Technical Note 1965-1

Programmable Film Reader

A. W. Armenti
D. F. Clapp
A. I. Schulman
R. A. Wiesen

10 March 1965

Prepared under Electronic Systems Division Contract AF 19(628)-500 by

Lincoln Laboratory
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
Lexington, Massachusetts

The work reported in this document was performed at Lincoln Laboratory, a center for research operated by Massachusetts Institute of Technology, with the support of the U.S. Air Force under Contract AF 19(628)-500.

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

LINCOLN LABORATORY

PROGRAMMABLE FILM READER

A. W. ARMENTI — *Group 28*

D. F. CLAPP — *Group 22*

A. I. SCHULMAN

R. A. WIESEN

Group 25

TECHNICAL NOTE 1965-1

10 MARCH 1965

LEXINGTON MASSACHUSETTS

ABSTRACT

The Programmable Film Reader, consisting of digital computer, magnetic tape units, CRT, and film transport with optical and electronic circuits, is a device for reducing radar A-scope film data to digital form. This is done by scanning selected portions of the film with a spot of light under program control. The relative amount of light passing through the film is measured by the device and reported back to the computer for processing.

A set of computer programs, called the Film Reading Program System, has been written for the PDP-1 computer and Programmable Film Reader. These programs will read films in three formats; A-scope traces, A-scope traces with fiducial marks and Project Radar A-scope traces. The amplitudes of the traces are sampled up to about 500 times and the digitized results written onto magnetic tape in IBM format. This report presents a description of the computer programs together with flow charts and listings. The reader is presumed familiar with the PDP-1 computer and the MACRO assembly language.

The latest modification to the system adapts it for use with the MIDAS assembly program and with a new high-speed magnetic tape system on the PDP-1 computer.

Accepted for the Air Force
Stanley J. Wisniewski
Lt Colonel, USAF
Chief, Lincoln Laboratory Office

TABLE OF CONTENTS

I. PROGRAMMABLE FILM READER

A. General

In many cases, radar data are recorded by making photographs of an A-scope display. This display presents an amplitude vs time plot of the signal received by the radar in a chosen range gate. Although these photographic records have been valuable for qualitative viewing, their use in the past as a quantitative recording medium has been limited because of the enormous task of reading thousands of traces manually. In the course of a study of data processing for the Air Force Ballistic Missile Re-entry Systems tests, the need was recognized for a rapid, automatic method of extracting data from these photographs, and recording them on magnetic tape in a form suitable for entry into a computer. Hence, the development of a device to accomplish this was undertaken.

The basic elements of the Programmable Film Reader, shown in Fig. 1, are the CRT which displays a point of light under program control; the reference photomultiplier which is adjusted so that its output is a constant voltage that is a measure of the background level of the film; the signal photomultiplier whose output voltage is a measure of the density of the part of the film illuminated by the point of light displayed on the CRT; the difference amplifier and threshold detector which determine whether trace or background was illuminated; the digital computer; and the magnetic tape units. The computer is used to move a spot of light over the film, in an appropriate manner, to record the output of the threshold detector, and to record the digital data on magnetic tape.

The film reading system described has been in use since March 1963, and in the ensuing six months the feasibility of rapid, automatic digitization of A-scope film has been clearly demonstrated. More than 60,000 radar traces have been digitized at a reading rate ranging from 3 to 12 seconds per trace. The reading rate is principally a function of the number of light points displayed. The number of light points that must be displayed is primarily a function of film quality, film format, and the number

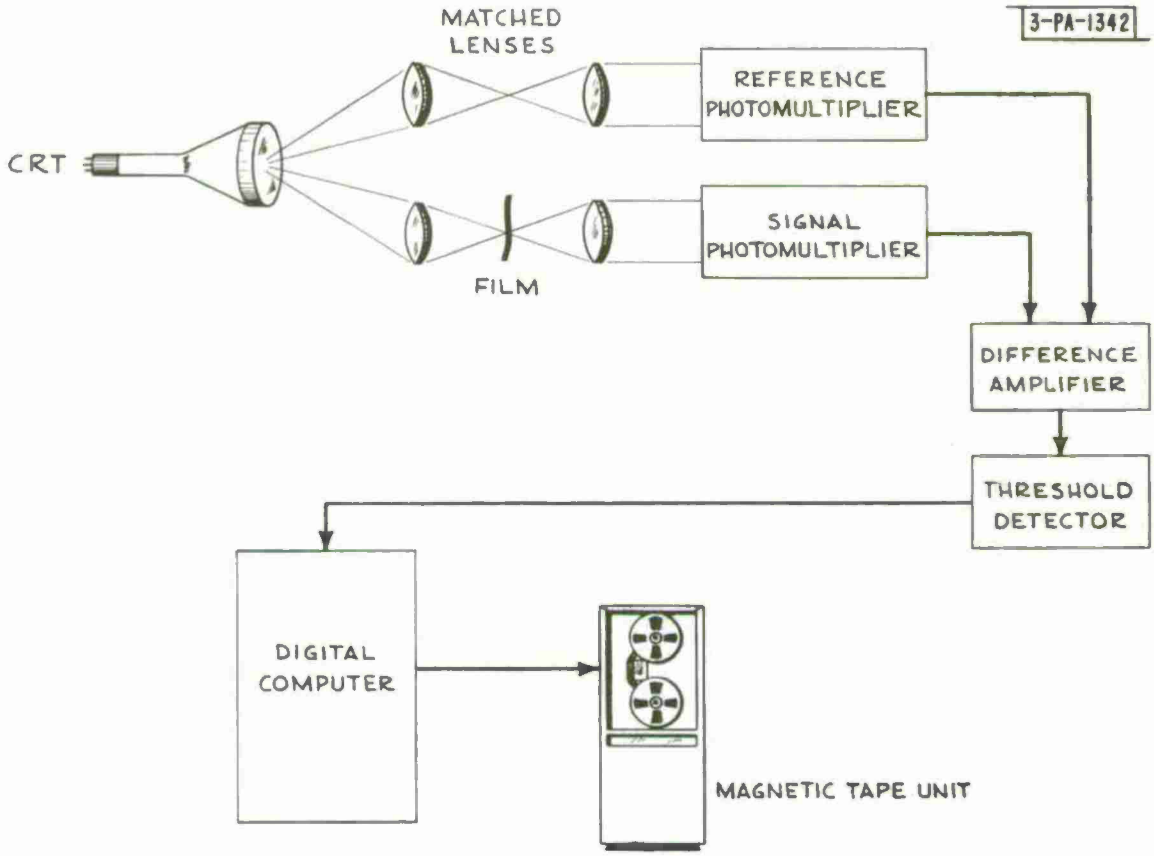


Fig. 1 Programmable film reader

of samples desired across the trace. For low quality film, e.g., film with appreciable photographic noise, a program is employed which applies a matched filter algorithm to the film reader outputs. This filtering process can require as much as 100 seconds reading time per trace. For high quality film, a program is employed which substitutes a simple selection algorithm for the matched filter routine. This program requires about 3 seconds reading time per trace. Because of this significant time saving obtainable with good quality film, a major effort has been made to improve film production procedures.

B. Equipment Components

The Lincoln Laboratory film reading system shown in Fig. 2 consists of four major pieces of equipment:

(i) a digital computer,

(ii) a CRT,

(iii) two magnetic tape units, and

(iv) the film transport and electro-optical comparison circuit.

The commercial versions of the first three equipment components currently in use are described in Digital Equipment Corporation publication # F-15D, entitled "PDP-1 Handbook", excerpted in Sections 1, 2, and 3 below.

1. Digital Computer

The digital computer used in this system* is a solid state, single-address, single-instruction, stored-program machine with five-megacycle circuitry, magnetic core storage with 5-microsecond access, and 18 bit parallel processing. As currently used, the machine consists of the central processor with a 4096 word memory module, (expandable in units of 4096 words to 65,536 words), a paper tape reader, a paper tape punch, and an on-line typewriter. The central processor contains the control unit, the arithmetic unit, the in-out transfer control, the memory module and a one-channel sequence break system.

---

* The PDP-1 built by Digital Equipment Corporation.

Fig. 2 Lincoln Laboratory film reading system

2. Cathode Ray Tube (CRT)

The CRT has magnetic focusing and deflection provided by solid-state circuitry. Using a 16-inch tube, it has a raster size of 9.25 x 9.25 inches with 1024 x 1024 addressable locations, of which 512 are resolvable along each axis. The display of each point requires 50 microseconds and is accomplished by means of an instruction, "Display One Point on CRT". The points are displayed with an accuracy of ± 3 per cent of the raster size.

3. Programmed Tape Control and Tape Transports

The computer in this system is equipped to use magnetic tape as an input-output medium by the installation of suitable magnetic tape controls and tape transports. The tape control transfers information between the computer Input-Output register and magnetic tape one character (7 bits) at a time. All transfer operations, including timing, formatting, error checking, and assembly of characters into computer words are performed by stored programs. The use of the tape control allows a choice of tape format, including the standard IBM format of 200 7-bit characters per inch, and inter-record gap of 3/4 inch and an inter-file gap of 3 1/2 inches.*

4. Film Transport

The film reader and transport mechanism used in the digitization consists of a twin-lens device built at Lincoln Laboratory for use on the TX-2 computer and later modified. Through electronic circuitry, the film reader can communicate with the computer.

The film reading equipment must do two things: (i) detect and indicate the passage of light through the film and (ii) advance the film under program control. The first task is carried out by the photomultipliers and electronic circuitry; the second task is accomplished under program control by the equipment described below.

---

*This system has now been changed to permit recording densities of 556 and 800 characters per inch (see Section III).

When all of the traces within view of the reader have been scanned, the program causes the film to advance by actuating a flip-flop in the computer (called "Program Flag 6"). The film is advanced by an 8-sprocket drive wheel turned by a Geneva drive which makes 1/4 revolution for every whole revolution of the driving motor. When flag 6 is on, the film advance motor runs, advancing the film by two sprocket holes per revolution. A cam, mounted on the motor shaft, in conjunction with a microswitch senses and reports to the computer every revolution of the motor shaft. In this way, the computer may turn on the motor, count its revolutions, and thus know how far the film will have been advanced.

C. Film Reading Program

In principle, the equipment described in this report could, with suitable computer programs, be used to read any signals or patterns that can be photographed in black and white. As currently implemented, the system reads A-scope photographs.

The logic of the program may be conveniently divided into four principal parts:

(i) the Set Up and Monitoring Routines,

(ii) the Vertical Scan Subroutines,

(iii) the Lateral Scan and Film Advance Routines, and

(iv) the Data Recording Subroutines.

The relation of these routines to the film reading process is shown in Fig. 3.

1. Set Up and Monitoring Routines

The Set Up and Monitoring Routines include a typewriter control routine which permits the film reading operator, by using the on-line typewriter, to write a title record on magnetic tape, advance the magnetic tape a given number of records, initiate scanning to locate the traces in view of the reader, determine and/or change the contents of parameter locations in the program, etc.

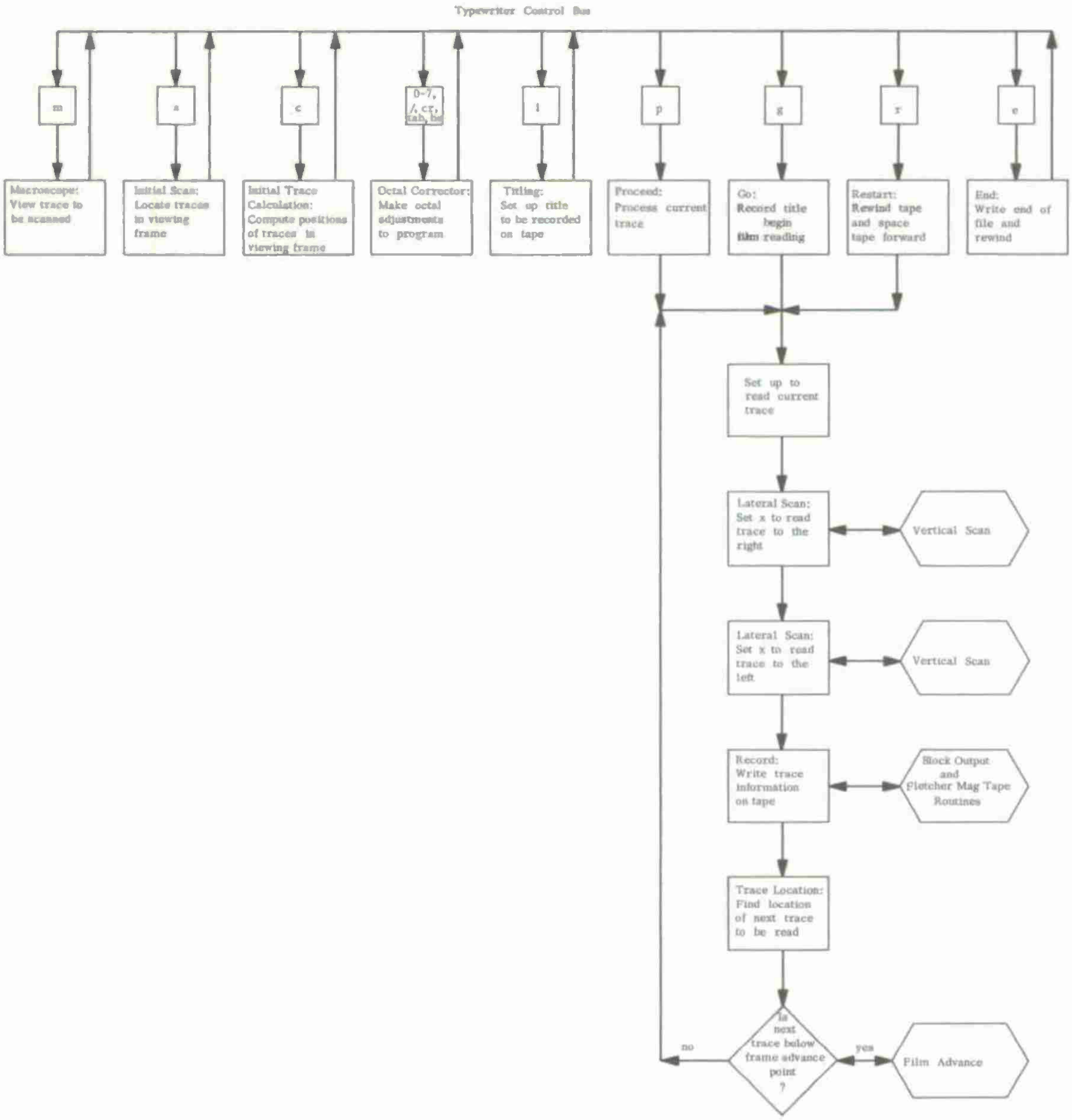


Fig. 3 General flow diagram of program operation

3-PA-1344

At the end of a successful read-in of the punched paper program tape, the program initiates a carriage return. At this point the operator may strike on the typewriter any one of 20 keys (l, 0-7, /, "tab", "carriage return", "backspace", s, c, g, p, e, r, m) which will be recognized by the program as a signal to transfer to a particular subroutine or program location; to any other character, the program responds with a question mark and remains in the typewriter listening loop. The characters recognized by the program when it is in the control loop and the action it will undertake when each character is typed are described below.

a. Titling Routine (l) Typing l causes the program to go to a titling routine. At this point the operator may type as many as 120 characters which will be accepted as a title to be written on magnetic tape. This routine causes the title to be stored in memory, the actual writing on tape being accomplished by striking "g" (see p. 10). The Titling Routine is terminated with a carriage return.

b. Octal Corrector Routine (Digits 0-7, /, tab, backspace, carriage return) Any one of these characters on type-in causes the program to transfer to a routine called the "Octal Corrector", the digits being interpreted as octal numbers. The Octal Corrector is used to determine and/or change the contents of a register in the computer. Typing in the four octal numbers designating a core location followed by a slash causes the routine to initiate a tab, produce a type-out in octal of the contents of that register, and initiate a second tab. After a type-out, if the operator wishes to replace the current contents with new contents, he may type in the new contents (in octal), and then strike the carriage return key. If the operator is satisfied with the register contents and wishes to make no change, he simply strikes the carriage return key, in which case the contents of the core location addressed by the last type-out will be typed out. This register may be changed by the operator, as described above. Use of tab permits, among other things, the insertion of program patches. After the initial type-out of the contents of a register, by typing "backspace" instead of "carriage return", the operator can cause the Octal Corrector subroutine to advance to the next location in sequence, thus avoiding the need to type each new core location.

After each "backspace", he may alter the contents of the register or not, as he desires. The "backspace" is useful for getting a type-out of the contents of a number of successive registers.

c. Initial Scan Routine (s) Typing "s" causes the program to transfer to a routine which initiates a vertical scan at an x-location set by the Test Word switches on the console. The vertical scan is used to locate the traces within a single viewing frame. The location information is used by other routines to determine when to initiate a film advance. For each point seen along the y-axis during the vertical scan a number is stored in a table whose register location corresponds to the x-position being scanned.

When the routine is in operation, the vertical scan line is displayed on the scope along with markers near the lower and upper ends of the scope showing the lower and upper limits respectively of the viewing frame used for film reading. Offset to the right of the vertical scan line, a point is displayed for every point seen on the film during the vertical scan.

During the operation of this routine, the operator can make optical adjustments to the signal and reference lenses until the offset points from the vertical scan line correspond to the number and positions of the traces on the section of film being viewed. This is a very critical part of the set-up prior to going into the automatic film reading mode since the initial settings of the trace positions will be used by the program to locate automatically the subsequent traces to be read.

d. Initial Calculating Routine (c) Typing "c" causes this routine to search the table set up by the scanning routine described in (c) above, and applies an exponential filter on the number of points seen. The location of a trace is the average of the position where the filter output rises above and falls below a threshold.

The location of each trace "seen", as defined by this process, is typed out if sense switch 4 is up. The positions typed out are the y-positions of each of the traces at the x-location set for the scanning routine. In addition to computing the estimated trace locations, this routine also computes tracking parameters to be used by the film advance routine in determining subsequent trace positions.

e. Go Routine (g) This character causes the program to transfer to a program location which writes the title record on the tape and starts the film reading process. As the film reading progresses through various phases, the program checks the computer console sense switch settings. There are six sense switches on the console that may be set by the operator. If sense switch 1 on the computer is turned on (i.e., the toggle switch is put in the up position), the program will repeatedly reread one trace without writing the data on the tape. Since the scanning will appear on the CRT, the operator can make fine optical adjustments to optimize the reading process. If sense switch 2 is on, all the traces within view of the reader will be read without writing the data on the tape. If sense switch 3 is on, the program will read all the traces in view and then cause the film to be advanced for the next set of traces without recording the data on magnetic tape.

If sense switch 6 is on, the program will cause a momentary display of the trace as read; and if sense switch 4 and 6 are both on, the trace displayed will be held on the scope.

f. Proceed Routine (p) Typing "p" causes the program to transfer to a program location a few instructions beyond the starting location of the Go Routine at which point the film reading process begins without writing the title record. The operator uses the "p" type-in whenever he wishes to resume reading after an interruption. The program assumes that the tracking parameters are in order. If they are not, the operator should first go through the "s" and "c" type-in routines before typing "p".

g. <u>End Routine (e)</u> Typing "e" causes the program to transfer to a routine which writes an end-of-file mark on the tape and then rewinds it.

h. <u>Restart Routine (r)</u> Typing "r" causes the program to transfer to a routine which rewinds the tape on which the digitized data has been recorded, spaces it forward a controlled number of records, then restarts the processing at the location transferred to by "p". The number of records to be spaced forward is held in a register labeled "rc". The restart is useful to the operator, if for some reason he wishes to reread a section of film. He can back the film in the reader, set the "rc" register to space forward the appropriate number of records and then, by striking "r", cause the program to resume film reading from that point.

The "rc" register should be set to a number which is the number of traces recorded on the tape plus one.

i. <u>Macroscope (m)</u> Typing "m" causes the program to transfer to a routine called Macroscope which serves as an aid to the operator in making approximate setting of the optics by scanning the area within the view of a raster 512 x 336 scope coordinates. The results of the scan are displayed immediately below or above the viewed area. By setting test word switches on the computer console, the operator can move the location of the scanning raster horizontally and/or vertically. This feature of the routine is useful in determining the trace boundaries on the x-axis.

After the initial procedure described above, the program is ready to read film automatically. Basic to the reading are the Lateral Scan Routine and the Vertical Scan Subroutines which are described below.

2. The Vertical Scan Subroutines

The actual digitization of the traces on the film is done by making a sequence of vertical scans across the face of the CRT. Each such scan is accomplished by a Vertical Scan Subroutine, two of which are in current use. In each case, the subroutine searches within preset upper and lower limits until trace points are found, but the procedures employed and the criteria for determining when the trace has been found differ considerably.

3. The Lateral Scan and Film Advance Routines

The right and left movement of the vertical scan is controlled by the Lateral Scan Routine. Under control of this routine, the first vertical scan is made at $x = x0$, a quantity determined by the operator, and all subsequent ones are made at $x = (x0 + nI)$, $n = 1, 2, 3, \ldots$, and I is an incrementing quantity also set by the operator. After the preset right hand x-limit has been reached, the same procedure is resumed at $x = x0$ with negative x-increments until the preset left hand x-limit has been reached. At this point, the program decides whether another trace is within view or whether the film should be advanced. At the completion of this scan procedure, a table called the "Signal Table" has been filled with the ordinates of the trace determined by the Vertical Scan Subroutine for each point sampled along the x-axis. The program may be adjusted to take as many as 512 vertical scans per radar trace.

When the scanning of a single radar trace has been completed, the film reader moves down the film to find the next radar trace. This is done under the control of the "trace location" portion of the Lateral Scan and Trace Location Routines. Specifically, when the end of a trace has been found, and all "raw" data have been recorded, the probable location of the next trace is computed. When it is necessary to advance the film to view the next radar trace, the motor is turned on. The last trace found is tracked as the film advances in order to determine how far the film has moved. The anticipated position of the next radar trace is updated to reflect the movement of the film.

4. Data Recording Subroutines

The subroutines write a 514-word record on magnetic tape. The first two words are the trace count, and a spare, and the remaining 512 words are the Signal Table. The subroutines are also used to place a title at the beginning and an end-of-file at the end of the magnetic tape.

II. FILM READING PROGRAM SYSTEM

The Film Reading Program System is a group of related computer programs written for the Digital Equipment Corporation (DEC) PDP-1 Computer in the MACRO assembly language and is meant to be used with the Lincoln Laboratory Programmable Film Reader, described in Section I.*

These programs, which now exist in three "series", are the result of an extensive revision of the so-called Radar Scan and Baseline Scan programs originally written for use with the film reader. The three "series" are designed to read films of three different formats (see Fig. 4):

Series 1 - A-scope traces on strip film,

Series 2 - A-scope traces on strip film with a single fiducial mark,**

Series 3 - Project radar film, framed film with A-scope traces, timing light and fiducial marks .

In addition, there are two modified programs, Series 1a and Series 2a, which are written to read traces whose orientation is upside down compared to those read by the Series 1 and 2 programs.

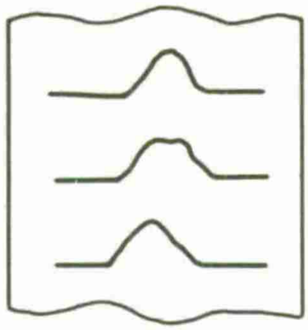
Since all of the programs of the system are similar in many respects, only the Series 1 program has been described completely, with the differences between it and the other programs being described in like detail. Textual description, flow charts and listings have been supplied for each.

---

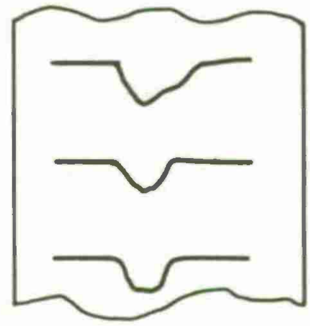
* Section I contains a general flow diagram (Fig. 3) of the original film reading programs as well as a brief description of each major program block.

** A zero-voltage reference mark at the beginning, and sometimes the end, of the trace; about one μsec long in a 10 μsec sweep.

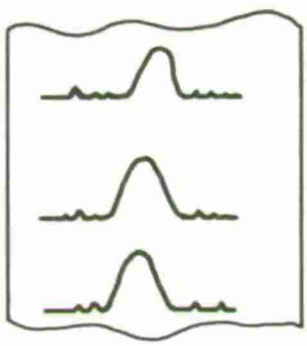
3-PA-2156



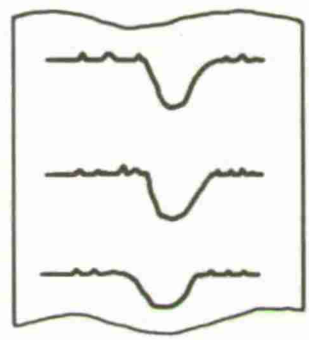
SERIES 1 FORMAT



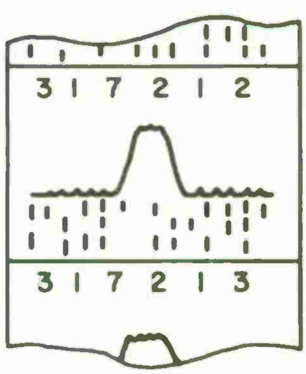
SERIES 1a FORMAT



SERIES 2 FORMAT



SERIES 2a FORMAT



SERIES 3 FORMAT

Fig. 4 Film formats

A. Series 1

The computer program described below is similar in many ways to the program described in Sect. I and corresponds closely to the general flow diagram given there as Fig. 3. The following will describe the various routines of the Series 1 program in some detail and should provide, in conjunction with the corresponding flow charts and listings, sufficient material to enable the reader to understand fully its operation.

The program operates most of the time in one of two states: it is in a two instruction "listening loop" in the typewriter control routine, or in a long loop in the lateral-scan control routine. In the former, the computer is waiting for instruction from the operator and in the latter, it is actually reading the film. There are other, less frequently used, excursions possible to which control may be switched from the typewriter and which lead back to the typewriter control routine. These will be described in turn.

```
film reading program system   series 1   25 feb 64 *

/definitions

define bintape
          dzm 677
termin

define bcdtape
          law 1
          dac 677
termin

define rewind
          jsp 5
termin

define space
          dac .+2
          jsp 14
          0
termin

define weof
          jsp 163
termin

s=1s
msm=iot 73
mcs=iot 34
mcb=iot 70
mwc=iot 71
mrc=iot 72

start
```

*The material on this "title page" of the program listing consists of a set of macro-instruction definitions and parameter assignments used by the assembly program in producing the binary program. The macros are used throughout the program and their definitions should be noted by the reader before attempting to follow the listings.

1. Typewriter Control Routine

The object of this routine is to provide a mechanism whereby the computer may pause in its operation, wait for a command from the operator via the on-line typewriter and then transfer control to other routines corresponding to typed commands. Its first instruction is located at beg= 1040 and is the first instruction performed after read-in of the program.

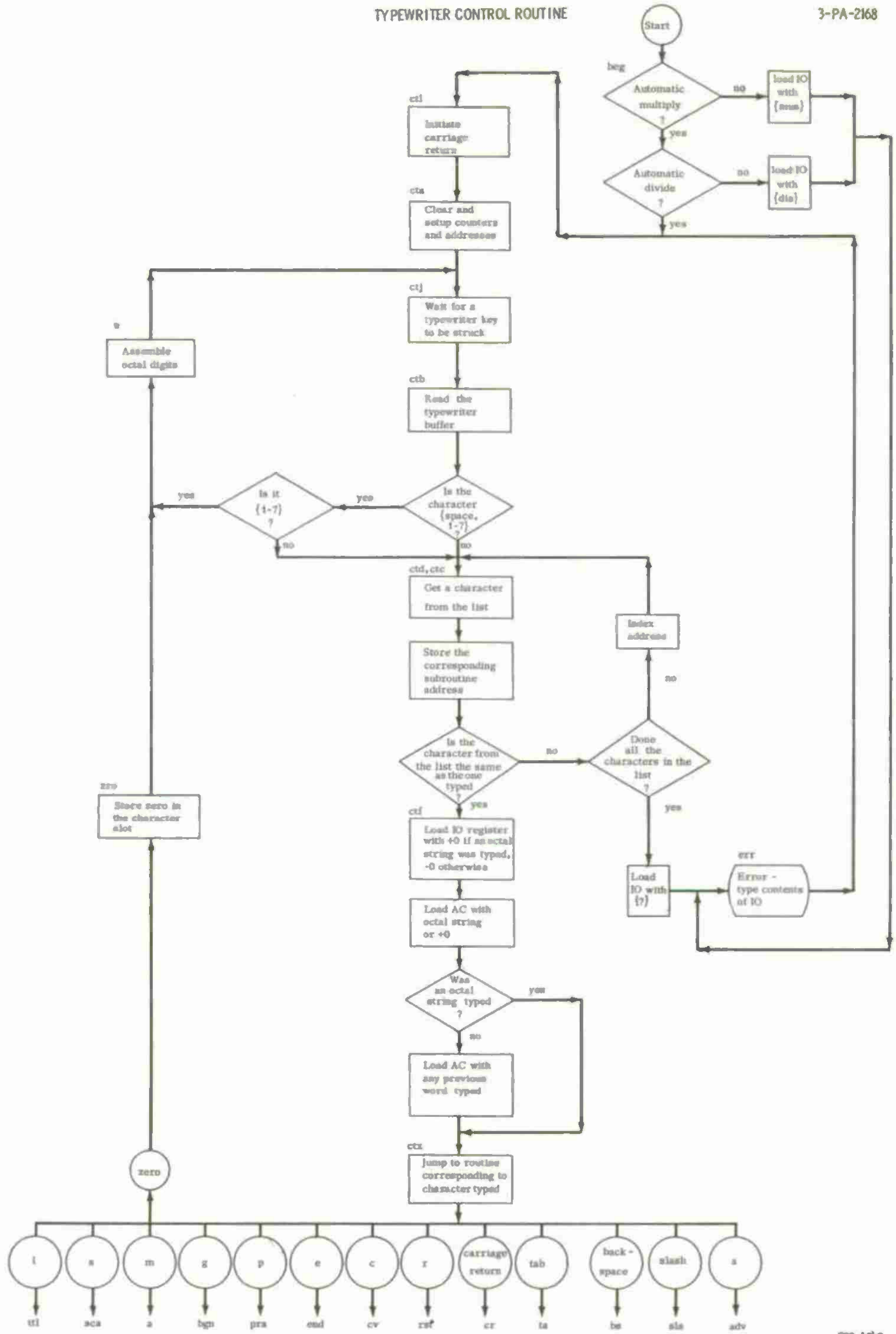
The so-called "listening-loop" is the pair of instructions at ctj+1 and ctj+2. On the way to this loop, the program checks to see that the automatic multiply and divide switches are set and, if not, causes "mus" and/or "dis" to be typed out. Then a carriage return is typed out, registers are initialized, the AC, IO and program flags are cleared and the machine enters the listening-loop, where it remains until a typewriter key is struck (i.e., program flag 1 is set to 1).

This version of the program recognizes the characters: a, c, e, g, l, m, p, r, s, carriage return, tab, backspace, slash and the octal digits 0-7. To all others, it responds with a typed out question mark. The digits 0-7 are assembled as an octal string in the register <u>wrd</u>, while the other characters cause a transfer of control to other parts of the program.

The transfer is accomplished by the use of a dispatch table (stored from dtb to dte-1). Each register of the table has in the left six bits the CONCISE code for one of the characters recognized by the program and in the right twelve bits, the address of the corresponding routine.

TYPEWRITER CONTROL ROUTINE

3-PA-2168



typewriter control routine   15 jan 64

```
1040/

beg,        law 100
            mus (0              /check for automatic multiply
            lio (flex mus
            sza
            jmp erp
            law 100
            cli
            dis (200
            nop
            lio (flex dis
            sza i
            jmp erp

ctl,        lio (77
            tyo

cta,        clc
            dac c̄hi
            dzm w̄rd
ctj,        cla cli 7-opr       /listen loop
            szf i 1
            jmp .-1

ctb,        cla cli 7-opr
            tyi
            rcr 6s
            dac c̄h
            sub (100000
            sma
            jmp ctd             /not a digit between 1 and 7
            add (070000
            sma
            jmp n               /it is a digit between 1 and 7
ctd,        init ctc, dtb       /not a digit between 1 and 7

ctc,        lac                 /search for control character
            dap ctx
            xor ctx
            sad ch
            jmp ctf
            index ctc, (lac dte, ctc

err,        lio (flex ?         /not a legal character
erp,        repeat 3, ril 6s    tyo
            jmp ctl
```

```
ctf,        lio chi             /dispatch on control character
            lac wrd
            spi
            lac Iwt
            jmp i ctx
ctx,        0

zro,        dzm ch
n,          lac ch              /digits
            ral 3s
            ior wrd
            ral 3s
            dac wrd
            dzm chi
            jmp ctj

dtb,        char ll ttl         /character dispatch table
            char ls sca
            char lm a
            char lg bgn
            char lp pra
            char le end
            char lc cv
            char lr rst
            char l0 zro
            770000 cr
            360000 ta
            750000 bs
            char l/ sls
            char la adv
dte,

start
```

2. Macroscope

When the computer is in the typewriter listening-loop and an "m" is typed, control is transferred to a routine called "macroscope" -- a program used to provide the operator with a view of what the film reader is seeing. To this end, a raster is displayed on the scope face and each point of it seen by the reader is redisplayed after being offset vertically to an area not covered by the raster. The program cycles until stopped (by setting flag 1 via the typewriter) and during this time the position of the raster is set by means of the test word switches: x controlled by positions 0-8 and y by positions 9-17. Thus the operator may view all portions of the film visible to the reader.

In order that the (apparent) cycle time will be reasonably short, the program is written so that the full cycle consists of sixteen sub-cycles, each of which displays a raster with only 1/16 the density of the full raster (which is made up of every other addressable location). To do this, the 512 x 336 point raster is divided into 64 x 42 boxes with eight points per side, and during each sub-cycle one point is displayed from each box. Incrementing is done in both directions by 2 points, so that the full cycle consists of display of 16 points in each box.

Thus, the program begins by displaying the lower left point of each box in the bottom row of the raster, redisplaying the points, relocated in the y direction, as they are seen by the reader. (The relocation is down if the initial y location from the test word is positive and up if negative.) Next, the program moves up one row of boxes at a time and displays the lower left point of each box. When the top row is finished (and displayed) the sub-cycle is over.

The next sub-cycle displays point (0, 1) of each box -- assuming that the lower left corner was (0,0) -- and the following sub-cycles display points (0,2) (0,3), (1,0), (1,1), (1,2), (1,3), (2,0), (2,1), (2,2), (2,3), (3,0), (3,1), (3,2), and (3,3), thus completing the display of the raster.

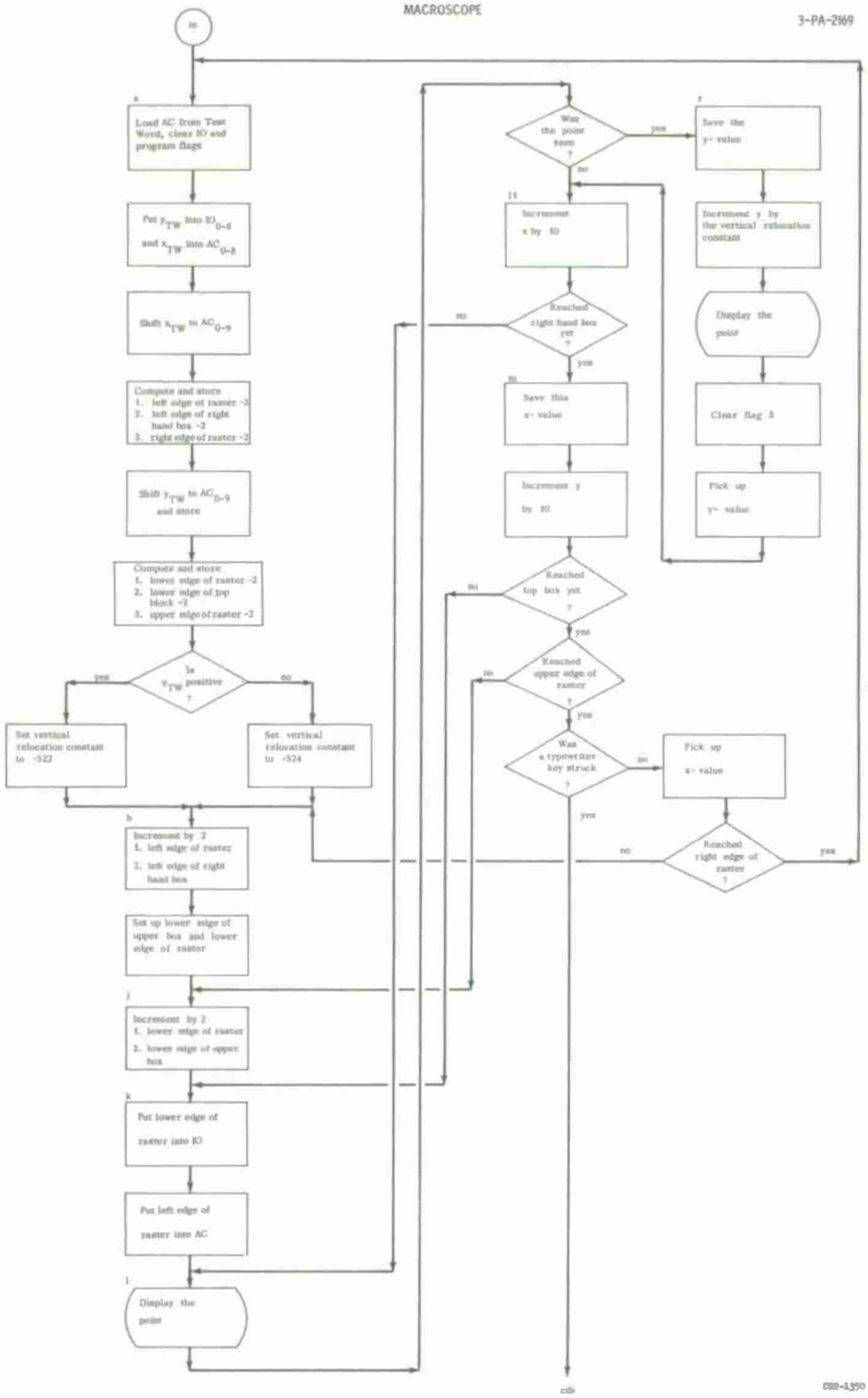
The coordinates entered in the test word are interpreted as the coordinates of the center of the raster. The use of only 9 bits for each coordinate implies that the maximum values of each are $\pm$ 377 (octal) or $\pm$ 255 (decimal). Thus, since the raster is 512 points wide, a maximum setting of the test word corresponds to the raster being at the edge of the scope in the x-direction, though not in the y-direction. Nevertheless, the range is easily enough to cover the whole area visible to the reader, which is restricted by the geometry of the system to about one-half of the area of the scope face.

The macroscope routine may be used to determine the left and right limits for the scanning process by moving the raster until an edge coincides with the limit of the desired scan. The test word switches may be read to get the x-coordinate of the center of the raster and the coordinate of the right edge will then be this number plus 400, while the left edge will be the number minus 400. The resulting numbers may then be used as c (<u>xmn</u>) and c (<u>xmx</u>) in the lateral scan routine.

The macroscope program may also be used to align the traces on the film with the axes of the scope. By watching the order in which points are redisplayed and tilting the reader left-to-right, the redisplay of a horizontal line may be made to occur during a single horizontal scan of the macroscope program. When this condition occurs the film horizontal and the scope horizontal coincide.

MACROSCOPE

3-PA-2169



```
macroscope  16 jan 64

a,          lat+cli 7-opr
            rcr 9s                /y to IO
            ral 9s
            sar 1s
            sub (200000+1000
            dac 1x̅0
            add (400000
            dac 1x̅1               /set up boundaries
            add (4000
            dac 1x̅m
            swap
            sar 1s
            dac t̅
            repeat 3, sar 2s      add t
            sub (125000+1000
            dac y̅00
            add (250000
            dac y̅11
            add (4000
            dac y̅mx
            lac t
            lio (-251777
            spa
            lio (252000
            dio v̅r

h,          law 1000              /horizontal interlace
            add 1x1
            dac 1x1
            xct h
            add 1x0
            dac 1x0
            lac y11
            dac 1y̅1
            lac y00
            dac 1y̅0

j,          xct h                 /vertical interlace
            add 1y1
            dac 1y1
            xct h
            add 1y0
            dac 1y0

k,          swap
            lac 1x0
```

```
l,          dpy
            szf 3
            jmp r
l1,         add (4000              /horizontal sweep
            sas 1x1
            jmp l

m,          dac t
            swap
            add (4000              /vertical sweep
            sas 1y1
            jmp k
            sas ymx
            jmp j
            szf 1
            jmp ctb
            lac t
            sas 1xm
            jmp h
            jmp a

r,          dio t                  /point seen
            swap
            add vr                 /vertical relocation
            swap
            dpy
            clf 3
            lio t
            jmp l1

start
```

3. Octal Corrector

The octal corrector routine provides a means to examine any register in core and change its contents if desired. If, when the machine is in the listening loop, an octal string of up to four digits* followed by a slash is typed, control is transferred to the octal corrector, which causes type-out of the contents of the register whose address was typed in (this process is called opening the register). The machine then types out a tab and returns to the listening loop.

The operator may then type in an octal string of up to six digits* which will replace the contents of the "opened" register. At this point he has three options:

(1) He may "close" the register by typing a carriage return, whereupon the new contents (if any) of the register will be stored and control returned to the listening loop. If no new contents were typed, control is simply returned to the listening loop.

(2) He may "close" the register as in (1) and then cause the next sequential register to be "opened" by typing a backspace.

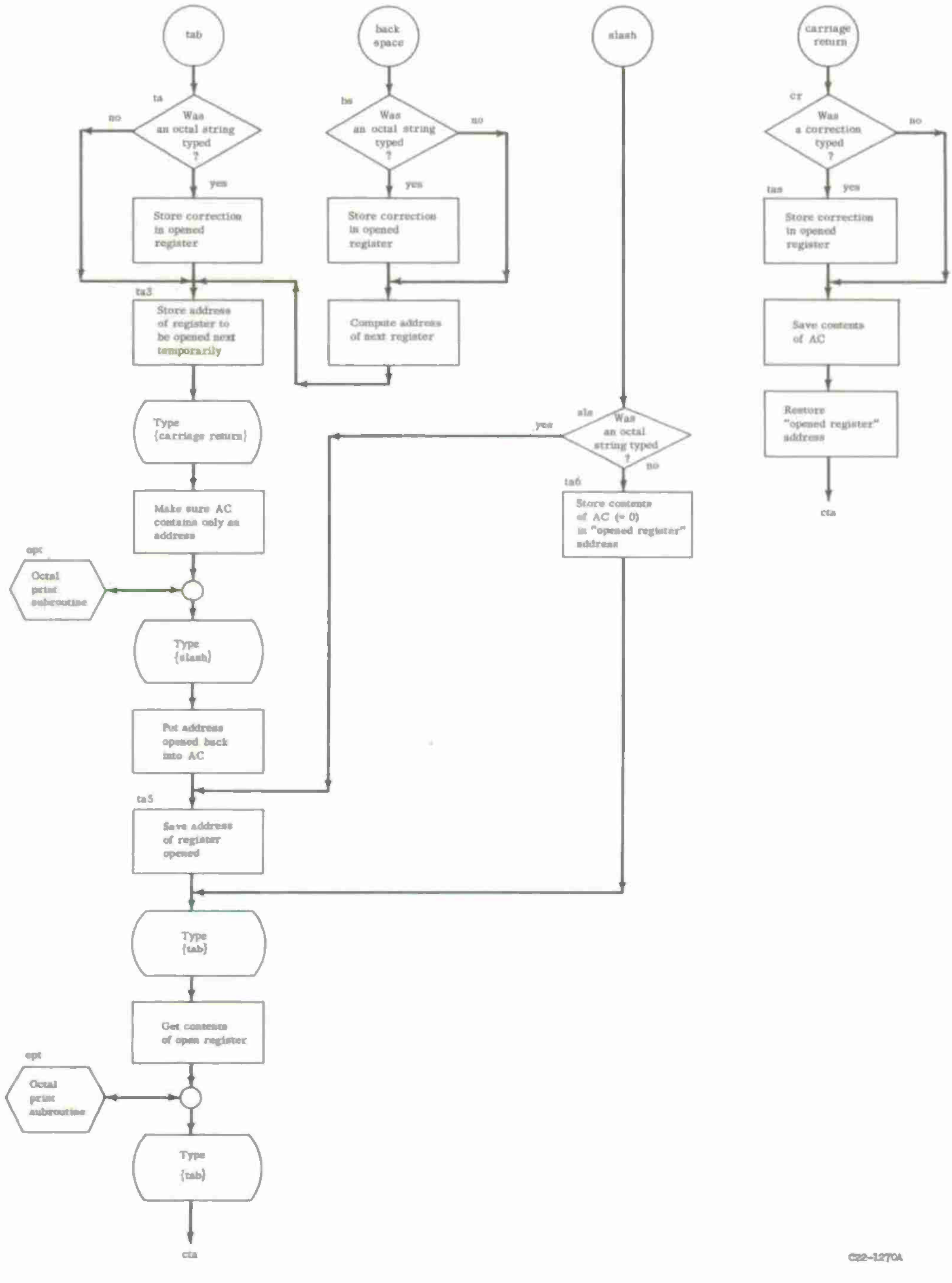
(3) He may "close" the register as in (1) and then cause the register addressed by that register to be opened by typing a tab.

---

* Leading zeros may be suppressed. If six digits are typed, the last four will be taken as the address. If more than six digits are typed, a word will be formed by performing successive Inclusive OR operations between the digits as they are typed and the previously formed word rotated left three places. E.g., typing the seven characters 3334445, leaves 334447 as the storage word; this is obtained by performing an IOR operation on the words 334443 and 000005.

OCTAL CORRECTOR

3-PA-2170



octal corrector 16 jan 64

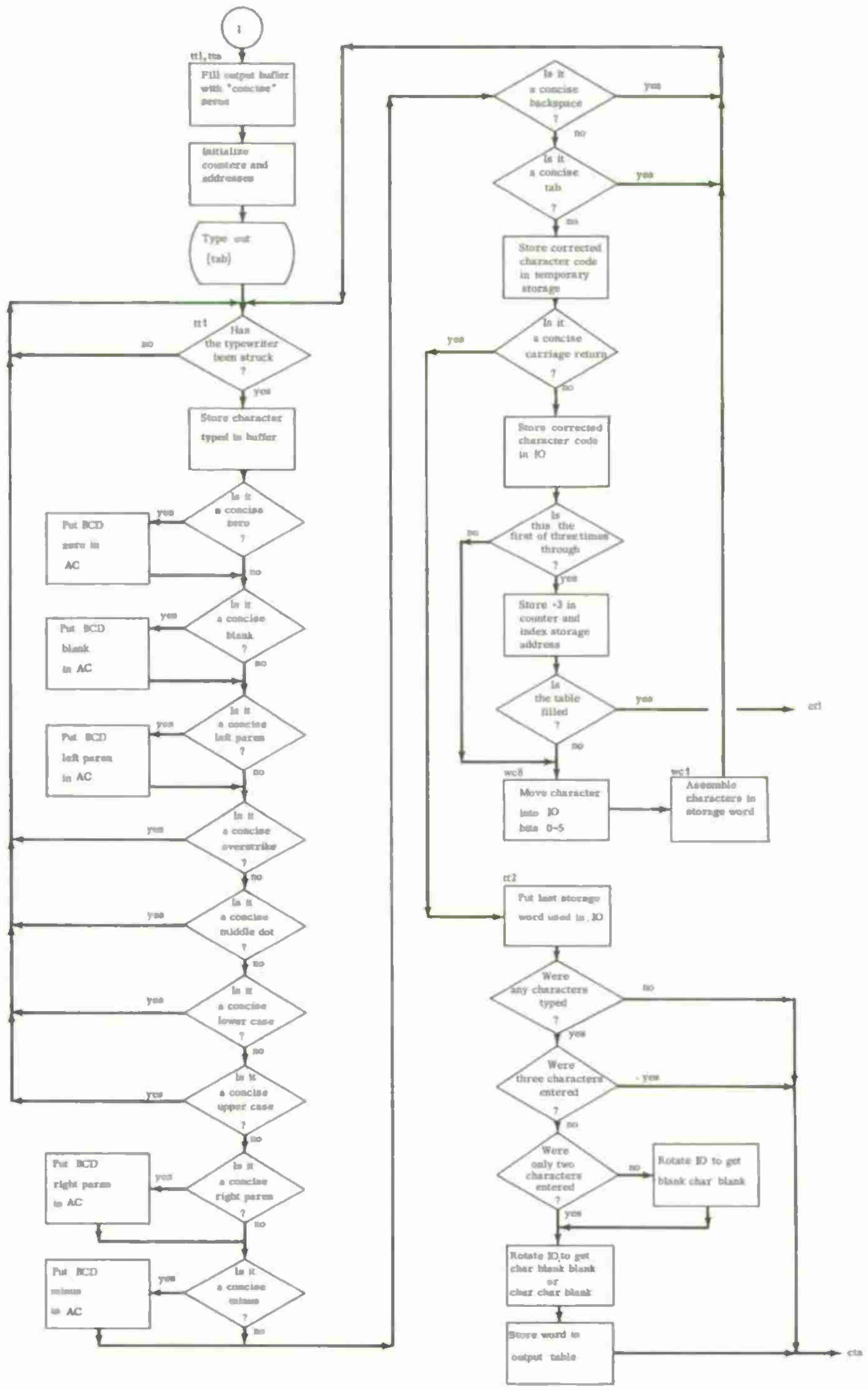
```
sls,        spi                  /was an octal string typed?
            jmp ta6              /no
            jmp ta5              /yes

bs,         spi i                /was a correction typed?
            dac i tas            /yes - store it
            idx loc              /no - compute address
                                 /of next register
            jmp ta3

ta,         spi i                /was a correction typed?
            dac i tas            /yes - store correction

ta3,        dac lwt              /store address as next
                                 /register to be opened
            lio (77
            tyo                  /type carriage return
            and (7777
            jda opt              /print address of register
                                 /to be opened
            lio (21
            tyo                  /type slash
            lac lwt

ta5,        dzm loc
            dap loc              /save address
ta6,        dap tas
            lio (36
            tyo                  /type tab
            lac i tas
            dac lwt
            jda opt              /type contents of opened register
            lio (36
            tyo                  /type tab
            jmp cta

cr,         spi i                /was a correction typed?
tas,        dac ch               /yes - store correction
            dac lwt              /no
            init tas, ch
            jmp cta

start
```

4. Titling Routine

The titling routine, to which control is transferred when the character "I" is typed, assembles up to 120 characters in a table of 40 registers after converting them from CONCISE code to IBM tape code. Those characters which have no IBM equivalent (e.g., overstrike, middle dot, backspace, etc.) are ignored. When the table is filled or a carriage return is typed, the computer returns to the listening loop.

TTLINC ROUTINE

3-PA-2171



C22-1351

titling routine 16 jan 64

dimension tbf(50)

```
ttl,        lio (202020          /initialize title table
            init tta, tbf
tta,        dio
            index tta, (dio tbf 50, tta
            dzm wcc              /initialize counter
            init wc1, tbf-1
            lio (36
            tyo                  /execute tab

tt1,        listen
            dio ch               /store character
            lac ch
            sad (20              /is char concise 0?
            law 12               /yes - put BCD 0 in AC
            sza i                /no - is char concise blank?
            law 20               /yes - put BCD blank in AC
            sad (57              /no - is char concise left paren?
            law 34               /yes - put BCD left paren in AC
            sad (56              /no - is char concise overstrike?
            jmp tt1              /yes - ignore char
            sad (40              /no - is char concise middle dot?
            jmp tt1              /yes - ignore char
            sad (72              /no - is char concise lower case?
            jmp tt1              /yes - ignore char
            sad (74              /no - is char concise upper case?
            jmp tt1              /yes - ignore char
            sad (55              /no - is char concise right paren?
            law 74               /yes - put BCD right paren in AC
            sad (54              /no - is char concise minus?
            law 40               /yes - put BCD minus in AC
            sad (75              /no - is char concise backspace?
            jmp tt1              /yes - ignore char
            sad (36              /no - is char concise tab?
            jmp tt1              /yes - ignore char
            dac ch               /store character
            sad (77              /is char concise carriage return?
            jmp tt2              /yes
            lio ch               /no
            isp wcc              /index char count - is result + ?
            jmp wc8              /no - char not first of new word
            law i 3              /yes - char is first char
            dac wcc              /reset character counter
            idx wc1
            sad (lac tbf 50      /is table filled?
```

```
        jmp ctl             /yes - title complete
wc8,    rir 6s              /no - move char into IO 0-5
wc1,    lac .               /bring contents of table register
                            /into AC
        rcl 6s              /put new char into AC 12-17,
                            /moving previous chars left six
        dac i wc1           /store chars back into table
        jmp tt1             /go listen for next character

tt2,    lio i wc1           /fix format of last table
                            /register used
        lac wcc
        sza
        sad (-1
        jmp cta
        sad (-2
        jmp tt3
        ril 6s
tt3,    ril 6s
        dio i wc1           /restore correctly formated
                            /register
        jmp cta


start
```

5. Initial Scan Routine

When the character "s" is typed, control is transferred to the initial scan routine, where it remains while the computer executes the instructions of a closed loop until another character is typed, whereupon control is transferred back to the typewriter control routine at ctb.*

The routine displays a vertical scan one point wide from the lowest addressable location on the CRT to the highest at an x-position determined by the ten left-most bits of the Test Word. The vertical scan begins at $y = 377777_8$ $(=400000_8)$** and advances in steps of two scope points (1000_8) until $y = 400001_8$. At this point the scan has passed the upper limit of the CRT raster.

As each point is displayed during this scan, its light will pass through the optical system of the reader and cause a signal to be sent to the computer or not depending on whether a trace point was "seen" or not. For each point seen, the routine displays a point on the CRT at the y-coordinate of the point seen and at an x-position offset to the right of the scan position by 40_8 points. This x-coordinate is computed at scc, where the quantity 20000_8 is added to the contents of the AC, corresponding to an increase in the x display coordinate of 40_8.

Furthermore, the number 1000_8 is stored in a slot in a table corresponding to the y-coordinate of each point seen. This table, cleared at the beginning of the routine at sca+1, extends from tbl, corresponding to $y = -777_8$, to tbl+777, corresponding to $y = 777_8$. The slot at tbl+377 corresponds to $y = \pm 0$.

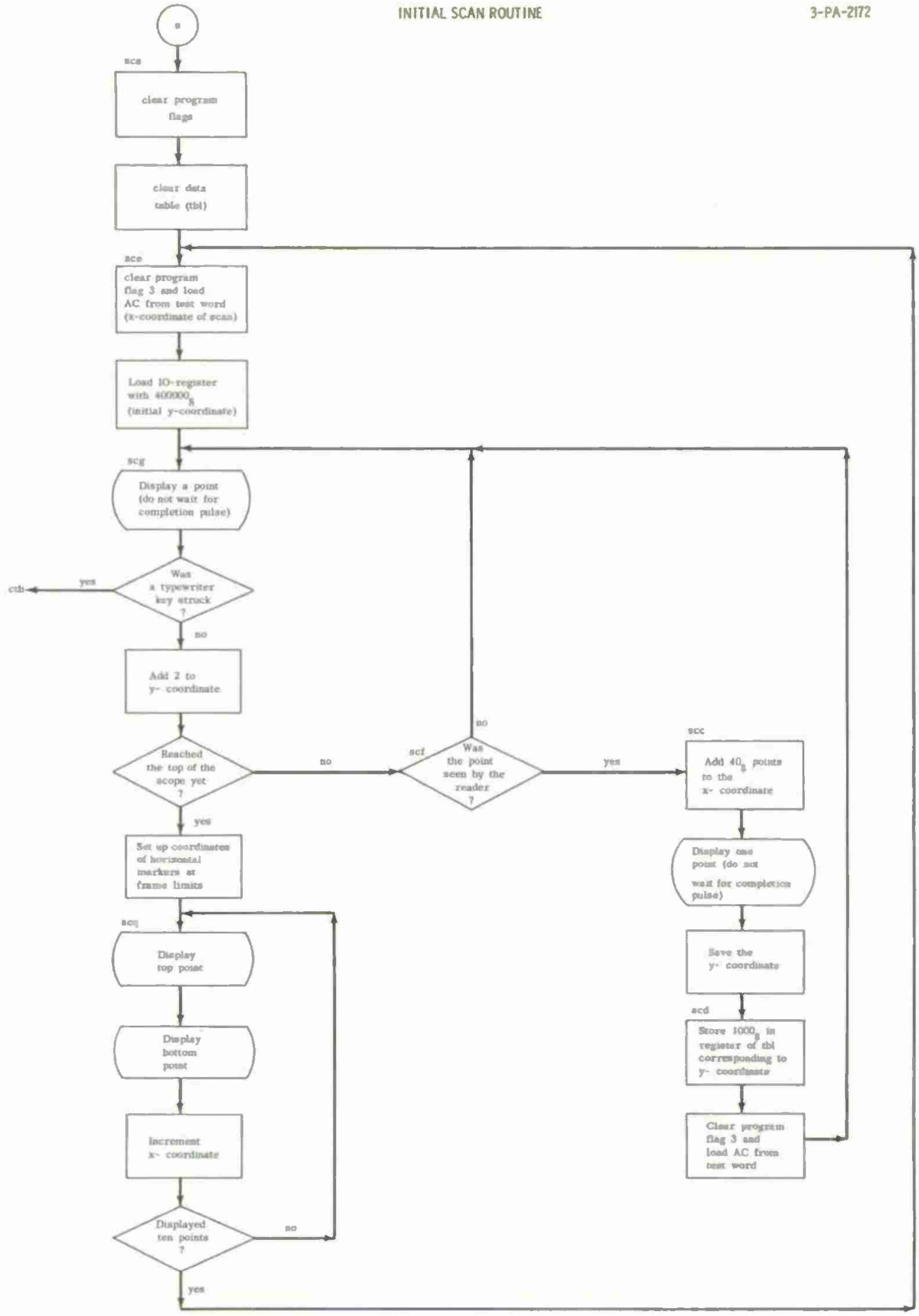
When the scan reaches the top of the scope, the routine computes the location of markers which show the upper and lower limits of the effective viewing area and displays them. The table of values and the computed limits are used by the initial position calculation routine described next.

---

*The character typed will be interpreted by the typewriter control routine.

**Negative numbers in the PDP-1 are represented by their one's complement. Thus any octal number with magnitude 400000 or larger is interpreted as a negative number.

INITIAL SCAN ROUTINE 3-PA-2172



C22-1272A

initial scan 16 jan 64

```
sca,        clf 7
            clear tbl, tbl 777    /clear data table
sce,        lat 3
            lio (400000
            jmp scg

scf,        swap
            szf 3                 /was the point seen ?
            jmp scc
scg,        dpy-i
            swap
            szf 1                 /was a typewriter key struck ?
            jmp ctb               /yes
            add (1000             /no - increment y
            sas (400001           /reached top of scope ?
            jmp scf               /no
            setup ch, 10          /yes - setup to display markers
            lac fmx
            sub fht
            sal 5s
            dac t1
            lat
            dac chn
scq,        lac chn
            lio fmx
            sil 5s
            dpy                   /display upper marker
            lio t1
            dpy                   /display lower marker
            add (1000
            dac chn
            isp ch                /done 10 points?
            jmp scq               /no
            jmp sce               /yes

scc,        add (20000            /point seen
            dpy-i                 /display point offset
            dio ch
            lac ch
            sar 9s
            add (tbl 377
            dap scd               /compute storage location
            law 1000
scd,        dac                   /store in table
            lat 3
            jmp scg

start
```

6. Initial Position Calculation

The initial position calculation routine computes the locations and spacing of the traces within view of the film reader as seen by the initial scan routine. The typing of the character "c" causes control to be transferred to this routine at cv, where, after preliminary housekeeping, the table prepared by the initial scan routine is used to compute the locations of the traces crossed by the scan.

The routine applies a simple exponential filter to successive entries in the table and then finds the y-values corresponding to pairs of successive crossings of a given threshold by the filter output. The trace locations are then taken to be the averages of these y-values.

The filter algorithm is

$$x_o = 0, \qquad x_i = \tfrac{1}{2}\,(v_i + x_{i-1}) \quad (i < 0)$$

where v_i is the i^{th} input value from the table, and

x_i is the corresponding filter output.

The input values, as mentioned earlier, are limited to the set $\{0,\ 1000_8\}$ corresponding to a point not seen or a point seen, respectively. The threshold value, c (thr), is currently set at 640_8.

The filter output is initially set equal to zero and, at cv1, the table entries are examined one by one and the filter outputs computed until the first pair of threshold crossings is observed. The y-values corresponding to these crossings then determine the location of the first, or lowest, trace (see Fig. 5a). The mean of these two values is stored at mm0, and a counter, initially zero, is indexed to show that exactly one trace has been found. The process is then restarted at cv1 with the next point of the input table.

When and if the routine detects a second trace, its location— found in the same way as the first —replaces the location of the first trace at mm0 and the difference between the locations is stored at mm1 (see Fig. 5b). The counter is indexed and processing continues.

The detection of a third or subsequent trace results in its location being stored at mm0, the last difference at mm1 and the difference between the last two differences (the last second difference) at mm2 (see Fig. 5c).*

If sense switch 4 is set as the routine is entered, the trace locations are printed out as they are computed.** The routine returns control to the typewriter control loop at ct1 when it has finished its pass through the input table.

* The value stored at mm2 is an acceleration term and does not show up on the drawings, of course.

** The values printed are not in scope coordinates, but rather are scope coordinates times 10_8.

3-PA-2157

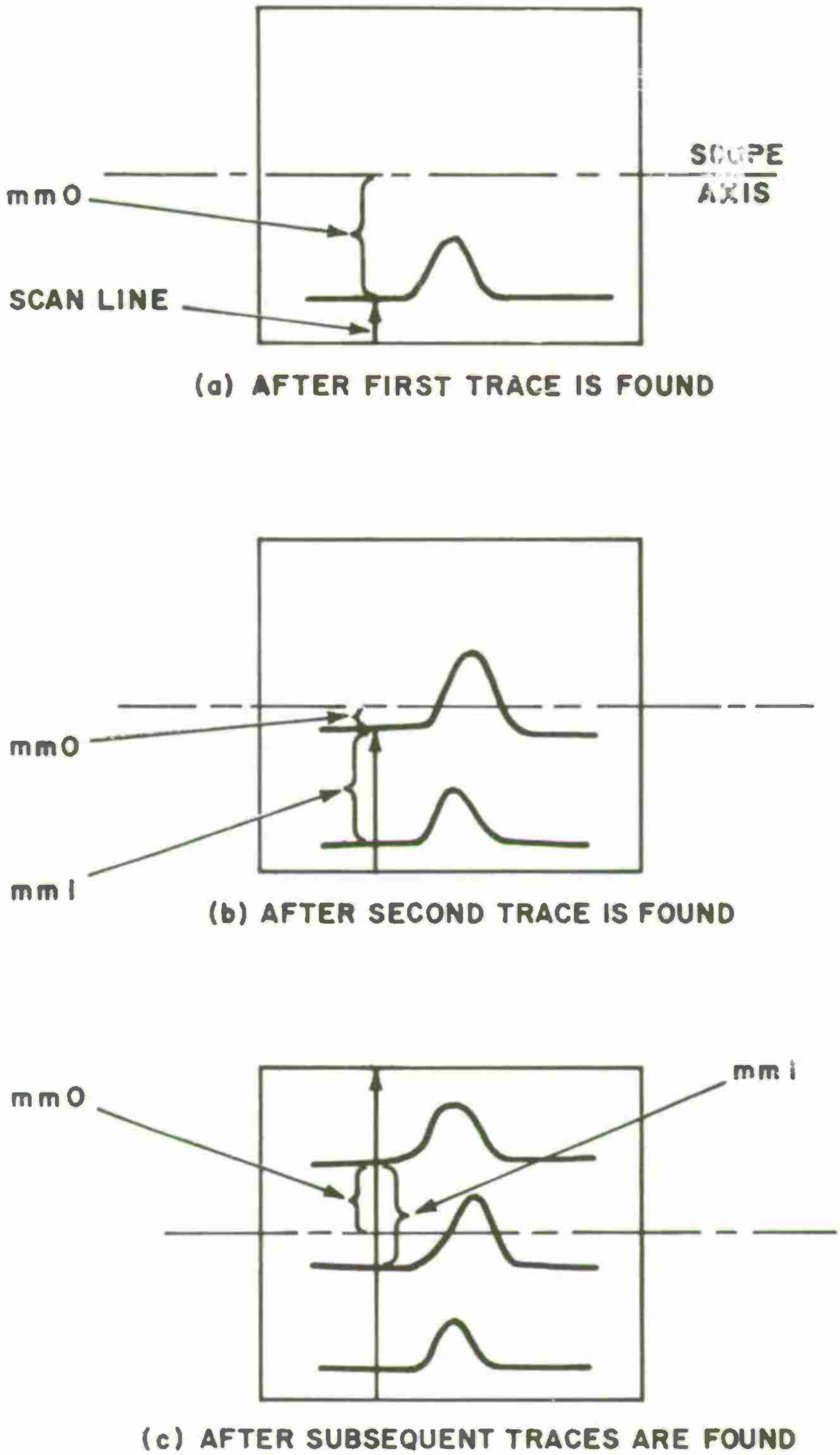
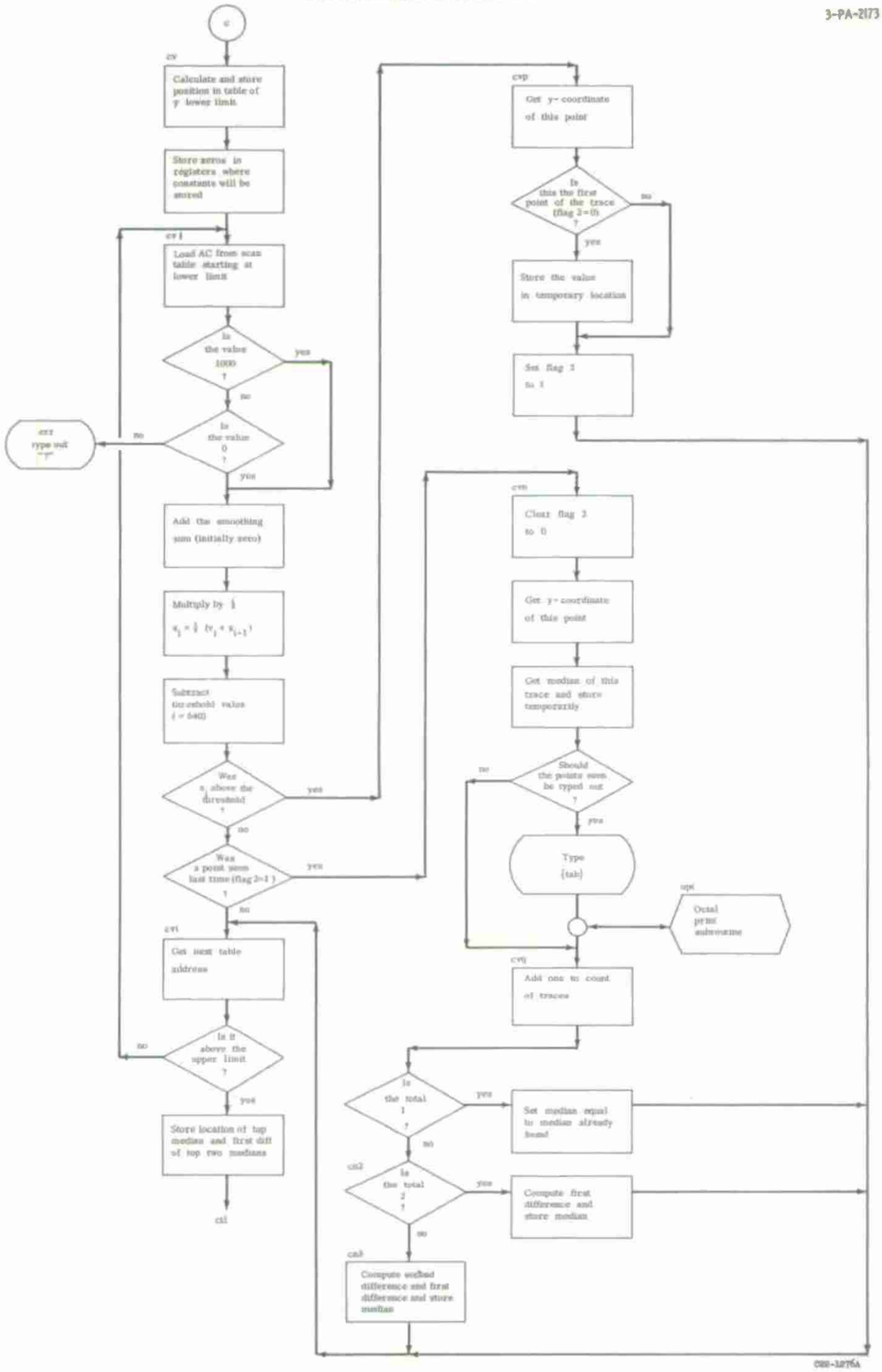


Fig. 5 Operation of initial position calculation routine

INITIAL POSITION CALCULATION ROUTINE

3-PA-2173



initial position calculation 16 jan 64

```
cv,        lac fmx
           sub fht
           sar 4s
           add (tbl 377
           dap cv1
           dzm c̄h
           dzm c̄hn
           dzm mm0
           dzm mm1
           dzm mm2
           dzm m̄t2
cv1,       lac .
           sas (1000
           sza i
           jmp . 2
           jmp err              /not data from scan program
           add ch
           sar 1s               /smoothing
           dac c̄h
           sub thr
           sma
           jmp cvp
           szf 2
           jmp cvn
cvi,       idx cv1
           sub (lac tbl 377
           sal 4s
           sub fmx
           spa
           jmp cv1
           lac mm1
           cma
           dac mm1
           dac mt̄1
           lac mm0
           dac mt̄0
           jmp ctl

cvp,       lac cv1              /above threshold
           sub (lac tbl 377
           szf i 2
           dac t1
           stf 2
           jmp cvi

cvn,       clf 2                /dropped below threshold
           lac cv1
           sub (lac tbl 377
           add t1
           sal 3s
           dac t̄1
           szs i 40
```

```
                jmp cvq
                lio (36
                tyo
                jda opt                         /type out positions

cvq,            idx chn
                sub (1
                szm
                jmp cn2
                lac t1                          /one trace
                dac mm0
                jmp cv1

cn2,            sub (1
                szm
                jmp cn3
                lac t1                          /two traces
                sub mm0
                dac mm1
                lac t1
                dac mm0
                jmp cv1

cn3,            lac t1                          /three or more
                sub mm0
                sub mm1
                dac mm2
                dac mt2
                add mm1
                dac mm1
                add mm0
                dac mm0
                jmp cv1


thr,            640                             /threshold


start
```

7. Lateral Scan Routine (Write, Title, Restart)

After the preliminary calculations have been made, control may be transferred to the lateral scan routine by means of one of three characters: g, r, or p. The character "g" causes control to pass to location bgn, where the title previously stored is written onto tape and zero is stored in counters which record the number of records on the tape, the number of traces read and the number of times the film has been advanced. Control then passes to pra.

The character "r" causes control to go to location rst, where the magnetic tape on which output is stored is caused to be rewound and then spaced forward the number of records given at location rc. Control then passes to pra.

The character "p" causes control to pass directly to pra, where the position of the first trace to be read is computed. In general, the program will start by reading the second trace from the top – the only exception being the case where only one trace is seen by the initial scan routine, in which case the program begins with that trace.

The routine then stores the x-coordinate of the left-most sample, c (xmn), in the output record and computes and stores the length of the output record based on the x limits – c (xmn) and c (xmx) – preset by the operator.

At location e1, the main loop begins with a check of flag 1 to see if a typewriter key was struck. If so, control is transferred back to the typewriter control loop, otherwise preliminary setup of limits and clearing of data table is done. At hs, an indicator, nps, is set to its initial value of unity; this is the "no point seen" indicator which is set to zero if a scan of a trace detects no points.

The program then does a single vertical scan at x = c (x0) between y = c (yll) and y = c (yul), by a jsp (Jump and Save Program counter) instruction to the vertical scan subroutine. This routine stores the y-coordinate of the trace found (if any) at location yf. Thus, the program has at this point the precise location of the trace which it will read at the x-location where reading will begin.

The next few instructions, up to e2, are designed to allow the program to read only every n^{th} trace, where n is a parameter which may be set by the operator. The instruction at skn is assembled as law i 1 and hence the program will read every trace without skipping any. Any other value may be substituted for the 1 by using the octal corrector.

At e2, the upper and lower scanning limits are recomputed on the basis of the location of the trace just detected.

Now begins the actual sampling of the trace amplitudes. In a six-instruction loop starting at hsa, the program causes a series of vertical scans to be made by the vertical scan subroutine at intervals of two scope units starting at x= c (x0) and continuing until x= (xmx). The x and y coordinates are reset to their original values and, at hsl, a seven-instruction loop repeats the scanning to the left from x= c (x0)-2 to x= c (xmn). Note that since the incrementing is by two units and the test at the end-points is one for equality, the numbers used as origin and left and right limits must all be even or all odd.

The vertical scan subroutine stores the amplitudes found in a table — called the signal table — located at tbl (=tba). This table forms a major part of the output record (see below).

At hsc, a test is made to see if any points at all were seen in the course of the scan: if none, the program will cause, at hsj, the characters "nps" to be typed and will then transfer control back to the typewriter listening loop. If at least one point was found, the routine continues to hst, at which point begins a section to find the mean of the amplitudes of the points found. As an additional safeguard, the program causes the characters "nls" to be typed out — and returns to the typewriter listening loop — if no amplitude values are found in the table of amplitudes. The mean value when found is stored at p2m.

At location rd, the routine checks the position of sense switch 6 to see whether or not it is desired to redisplay the contents of the signal table. If not, control is transferred to location bg, but if so, the routine sets up a counter, t3, to control the number of times the display will be cycled. This number can be changed easily by means of the octal corrector;* as assembled, it is ten (decimal). After displaying the trace this number of times, the routine checks sense switch 4 to see if the display should be held on the scope, i.e., whether or not the display should be recycled ten more times.

When and if sense switch 4 is off, control passes to location bg, the beginning of a routine which subtracts from each amplitude value measured the value of the average previously computed and stored at p2m.

This done, sense switch 1 is checked to see if the same trace should be rescanned: if yes, control passes back to e1; if no, the value of c (ym) is saved at pm and sense switch 3 is checked to see if the data for this trace should be recorded on tape.** The recording is done during the section which starts at ca, so that if the recording is to be suppressed, this section is skipped and control goes to location e.

At ca, the trace count, c (pc), is increased by one and then a checksum for the output record is computed by combining the contents of all of the registers of the output table (except the checksum itself) by an exclusive OR operation. This checksum is then stored at xyz in the output table.

This done, the length of the record to be written is computed and the calling sequence for the block-output routine is set up. This calling sequence consists of three instructions:

```
law   A
jda   735
law   B
```

where A is the address of the first register and B is the address of the last register of the output block. Flag 3 is then cleared, since it is used as an error indicator in the Fletcher Magtape Routines and could conceivably be set at this point.

---

* Change rd+3 from law i 12 = 710012 to law i n = 7100XX, where XX is the (octal) number of cycles desired.

** Note that sense switch 1 being set also suppresses recording by bypassing this section.

A check is then made of sense switch 5 to see whether the program is being used to read a block of n registers, n being a preset value stored at nt. If so, a check is made to see if n traces have been read yet, and if so, control is passed to the typewriter control routine at ctl. If neither of these, control passes to e, where the location of the next trace is estimated.

The trace just scanned passes through the point with coordinates: c(x0), c(pm). Location mm0 contains the position of the preceding trace and mm1 contains a (smoothed) estimate of the spacing between traces. Figure 6a shows the situation at the beginning of the computation. The equations used are:

$$c\,(\underline{mm1})_{new} = 7/8\ c\,(\underline{mm1})_{old} + 1/8 \quad c\,(\underline{pm}) - c\,(\underline{mm0})_{old}$$

$$c\,(\underline{mm0})_{new} = c\,(\underline{pm})$$

$$c\,(\underline{nwm}) \qquad = c\,(\underline{mm0})_{new} + c\,(\underline{mm1})_{new}$$

Figure 6b shows the situation after the computation.

The c(nwm), i.e., the estimated location of the next trace, is checked against a pre-stored limit, c(fap). If the next trace is expected to be below the limit – at x = c(x0) – then control is passed to the film advance subroutine. This routine moves the film in such a way that the image on the CRT of a point on the film moves up.

When control returns from the film advance subroutine the corrected (in view of the film movement) estimate of the location of the next trace is in the accumulator. It is checked against the lower limit again and the film is advanced until the trace is expected to be above the limit. When this happens, control moves back to e1 in order to scan the trace.

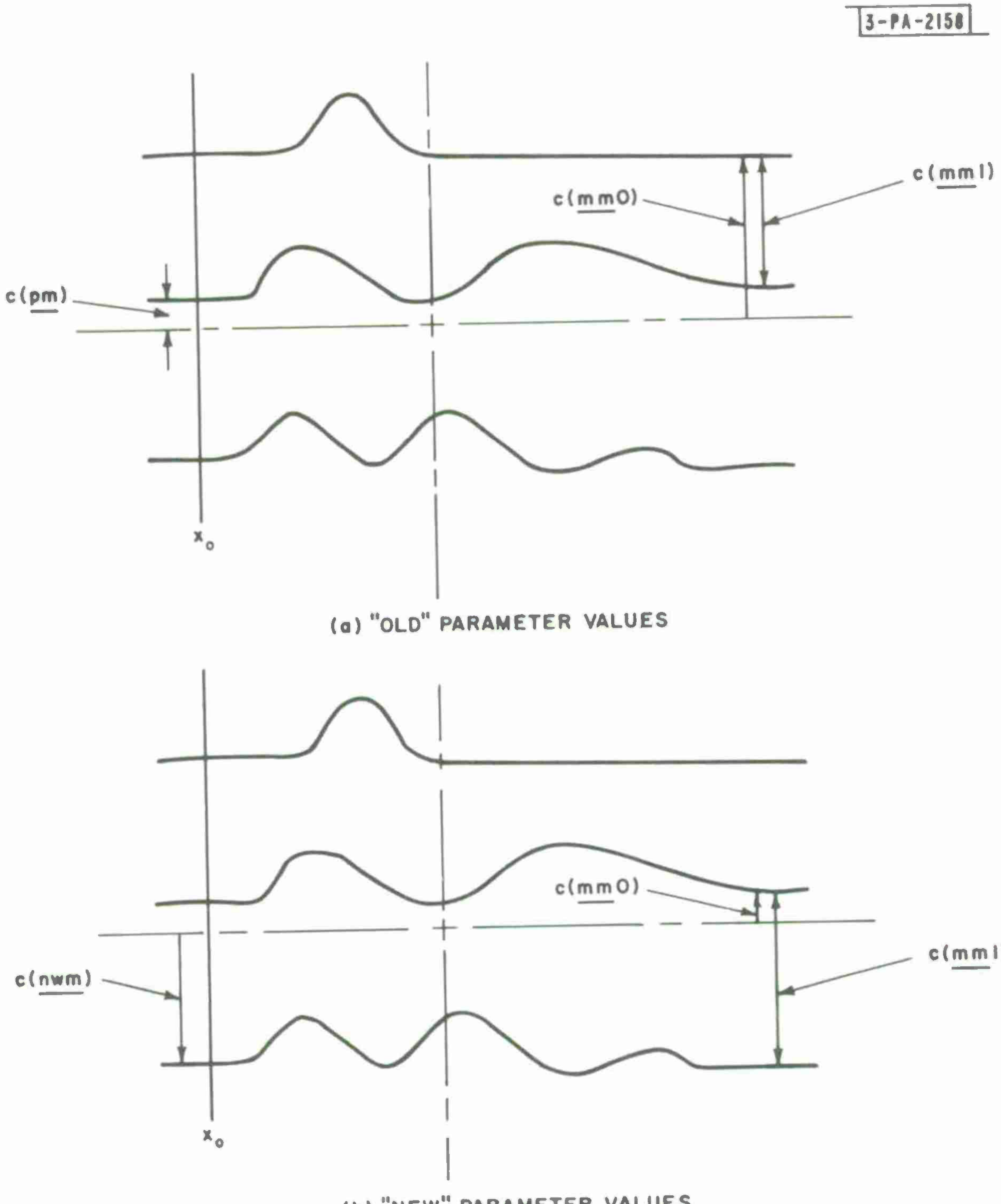
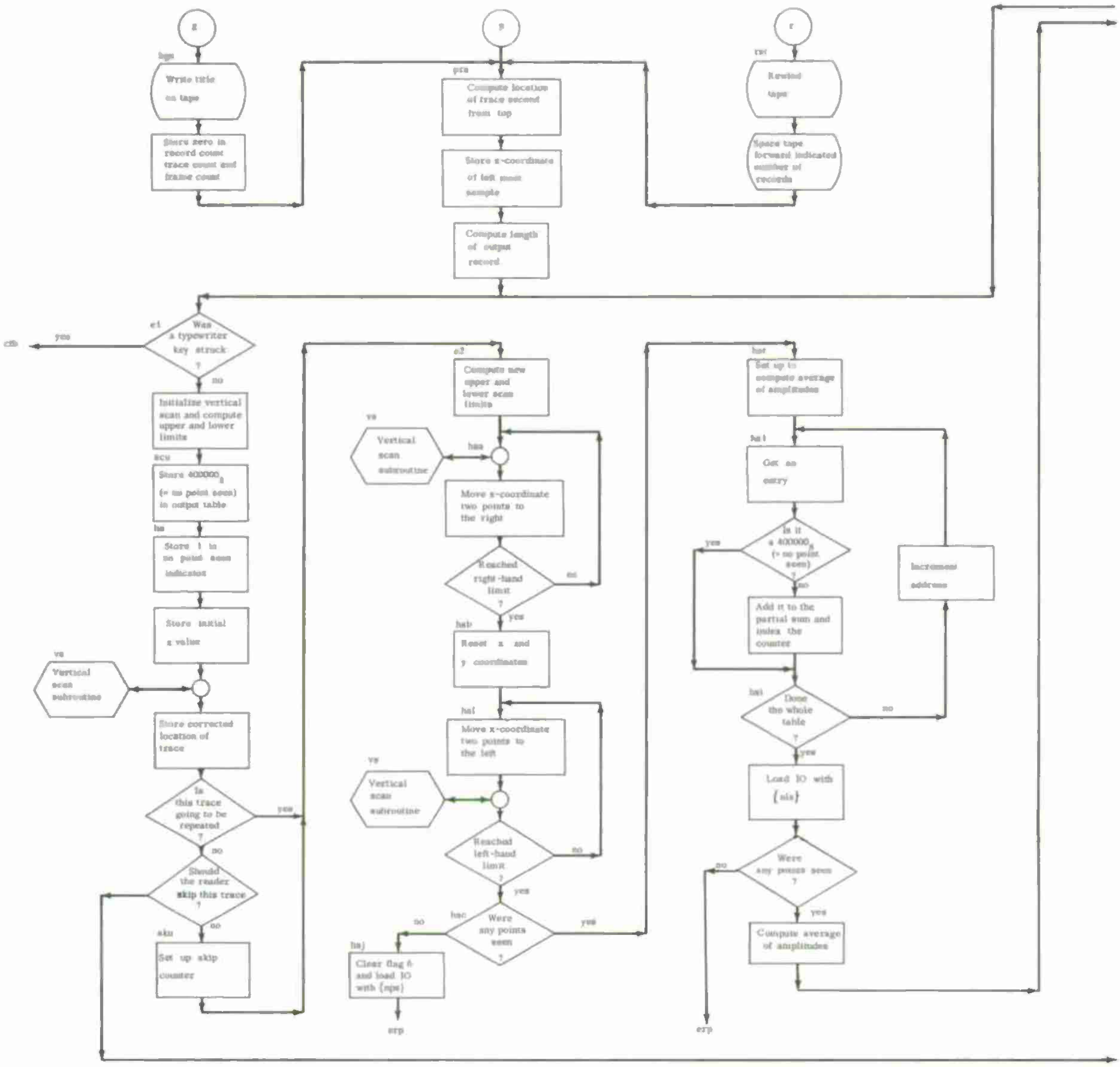


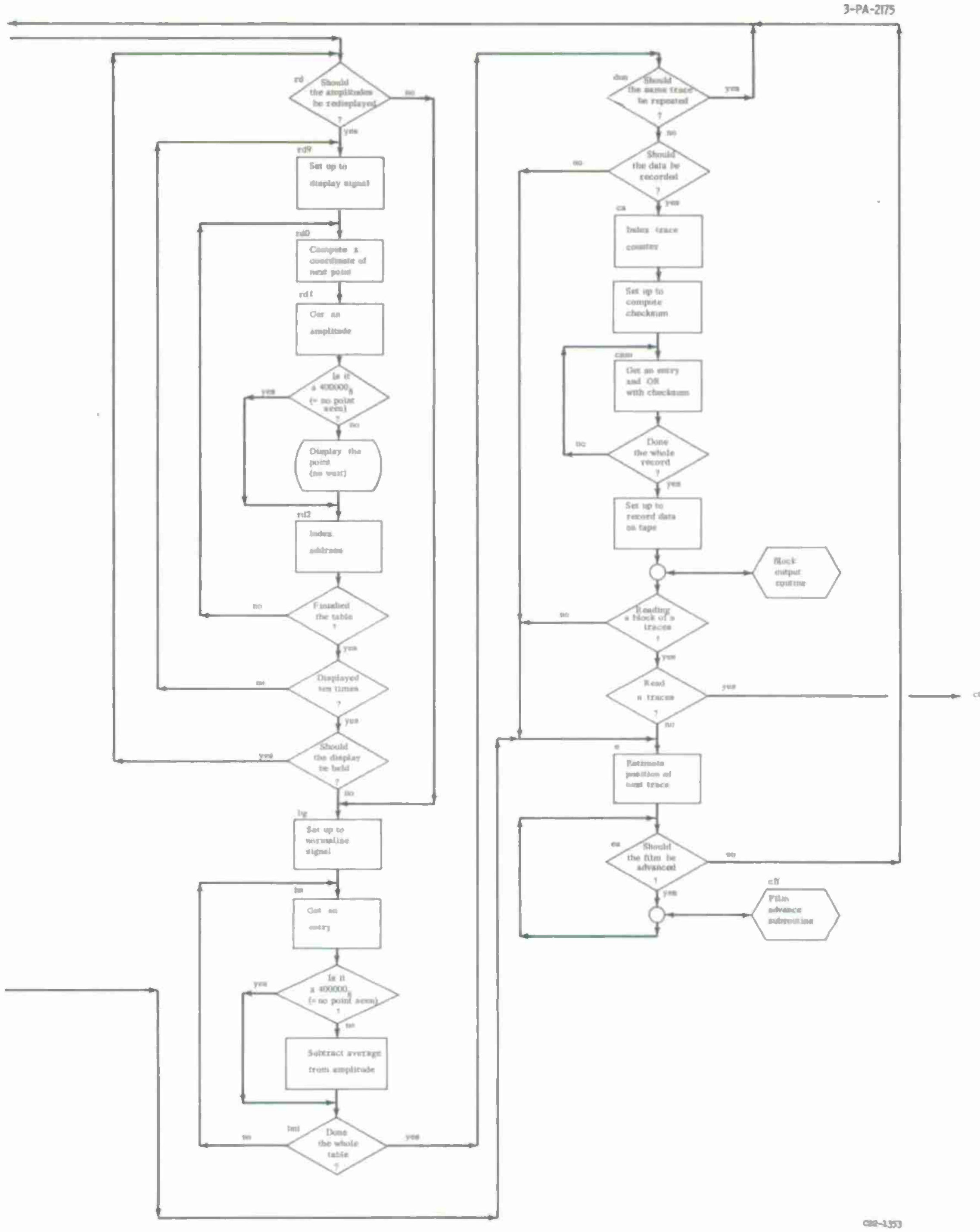
Fig. 6 Representation of parameter values before and after computation

3-PA-2174



LATERAL SCAN ROUTINE

3-PA-2175



C82-3353

write title - restart   3 feb 64

```
/write title

bgn,        bcdtape
            jsp 51
            tbf
            tbf 50
            -0
            opr
            bintape
            dzm rc              /record count
            dzm pc              /trace count
            dzm frc             /frame count
            jmp pra


/restart

rst,        rewind
            lac rc
            space
            jmp pra


start
```

lateral scan routine   25 feb 64

```
pra,        lac mm0
            add mm1
            dac nwm
            lac xmn             /store left-most x in output table
            sal 8s
            dac xst
            lac xmx             /compute length of output record
            sub xmn
            sar s
            add (2
            add csz
            dac rsz
            dzm sc

e1,         szf 1
            jmp ctb
            lac nwm
            sal 5s
            dac y0              /initialize vertical scan
            dac yf
            dac ym
            sar 8s
            add yvu
            dac yul
            sub yvu
            sub yvd
            dac yll

            law tbl
            dap scu
            lio (400000
scu,        dio .               /clear tables to nothing seen
            index scu, (dio tbe, scu

hs,         law 1
            dac nps
            lac x0
            dac x
            jsp vs              /find trace to refine position
            lac yf
            dac ym
            sar 5s
            dac pm
            szs 10              /should the same trace be repeated?
            jmp e2              /yes
            isp sc              /no
            jmp e
skn,        law i 1             /(-1)(number to skip + 1)
            dac sc
```

```
e2,         lac yf
            sar 8s
            add yvu
            dac yul
            sub yvu
            sub yvd
            dac yll
hsa,        jsp vs
            law 2
            add x                   /right
            dac x
            sas xmx
            jmp hsa

hsb,        lac x0                  /reset to go left
            dac x
            lac ym
            dac yf

hsl,        law i 2                 /left
            add x
            dac x
            jsp vs
            lac x
            sas xmn
            jmp hsl

hsc,        lac nps                 /done
            sza                     /anything seen?
            jmp hsj                 /no, complain

hst,        init hs1, tba           /calculate average as first
            dzm t1                  /approximation to baseline
            dzm p1m

hs1,        lac
            sad (400000
            jmp hsi
            sar 8s
            add p1m
            dac p1m
            idx t1

hsi,        index hs1, (lac tbe, hs1
            lio (flex nls
            lac t1
            sza i
            jmp erp
```

```
                lio p1m
                cla
                spi
                clc
                scl 1s
                dis t1
                hlt
                dac p2m

rd,             szs i 60
                jmp bg
                setup t3, 12

rd9,            init rd1, tba     /redisplay signal

rd0,            lac rd1
                sub (lac tba
                sal s
                add xmn
                sal 8s
                dac t1
rd1,            lac
                sad (400000
                jmp rd2
                swap
                lac t1
                dpy-i

rd2,            idx rd1
                sas (lac tbe
                jmp rd0
                isp t3
                jmp rd9
                szs 40
                jmp rd

bg,             init bn, tba
bn,             lac
                sad (400000
                jmp bni
                sar 8s
                sub p2m               /normalize signal
                sal 8s
                dac i bn
bni,            index bn, (lac tbe, bn

dun,            szs 10                /should the same trace be repeated?
                jmp e1                /yes
                lac ym                /no
                sar 5s
                dac pm
                szs 30                /should the data be recorded?
                jmp e                 /no
```

```
ca,         idx pc              /yes - index trace counter
            dzm xyz             /store zero in checksum register
            dzm ch
            law pc              /set up to compute checksum
            dap csm
            add rsz
            dap pnd
csm,        lac                 /get an entry
            xor ch              /compute checksum
            dac ch
            idx csm
            sas pnd             /done the whole record?
            jmp csm             /no
            lac ch              /yes - store the checksum
            dac xyz
            law pc
            add rsz
            sub (1
            dap . 3
            law pc              /record data on tape
            jda 735
            law
            clf 3
            szs i 50            /reading a block of n traces?
            jmp e               /no
            lac pc              /yes
            sub nt
            sma                 /read n traces?
            jmp ctl             /yes

e,          lac pm              /no - estimate new median
            sub mm0
            sar 3s
            dac t2
            lac mm1
            sar 3s
            cma
            add mm1
            add t2
            dac mm1
            lac pm
            dac mm0             /constant velocity assumed
            add mm1
            dac nwm             /best guess for new median
ea,         sub fap             /frame advance point
            sma
            jmp e1
            jsp cff
            jmp ea

hsj,        clf 6
            lio (flex nps
            jmp erp

start
```

8. Film Advance Subroutine

When, following the scanning of a trace, it is determined that the next trace will be "out of sight", i.e., the computed location at which scanning should start is below the preset lower limit, control is shifted to the film advance subroutine to move the film up and bring the next trace into view.

The routine begins at cff and, after saving the return address, checks sense switch 2 to see whether or not it is desired to reprocess the traces in view rather than advance the film. If it is, control jumps to location cfh where the values of c (mm0), c (mm1) and c (mm2) computed during the last running of the initial position calculation routine are restored. Thus, the lateral scan routine will repeat the scan of the traces in view if the film has not been advanced since the last run of the initial position calculation routine, or if the values of c (mm0) c (mm1) are by coincidence appropriate to the current configuration. If the film is to be advanced, the x and y coordinates of the last trace are stored and then the film advance motor is turned on by setting program flag 6 to 1.

As described earlier, there is, on the shaft of the film advance motor, a cam-microswitch combination whose status may be determined by the computer (see Fig. 7). The film advance routine repeatedly checks the micro-switch position by executing the

-PA-2159

switch "on" — iot 11 makes $io_4 = 0$

switch "off" — iot 11 makes $io_4 = 1$

Fig. 7 Cam-microswitch combination

iot 11 instruction and examining the IO register until it sees the sequence "off-on-off" or "on-off", whereupon it stops the film advance motor by clearing program flag 6. Between each interrogation of the microswitch, control is passed to another subroutine— the trace following subroutine, to be described below — to locate and record the position of the moving trace. A counter, frc, is used to keep trace of the number of times the motor is turned on and off.

When the film has stopped moving and the motor has stopped turning, the routine selects the last known position of the trace, c (yf), and initiates a single vertical scan in its vicinity. The newly computed trace position replaces the previous c (yf).

The distance the film moved can then be computed as

c (fav) = c (yf) - c (ym)

where ym will contain the y-coordinate of the trace before the film was moved. This amount is used to update the contents of mm0 and nwm, and control is returned to the main program.

The trace following subroutine is a closed subroutine called at cfa and cf1 which keeps track of the location of the trace last read while the film is being advanced. The routine begins, after it has checked to see if a typewriter key has been struck, by setting up the upper limit of the scan — to c (yf)+c (k2) — and setting the threshold to 5/13.

The plan is to display a band of points — 13 points wide — starting below the trace and extending vertically until the trace is found (see Fig. 8) The program is able to display the band so fast that it can easily keep up with the advancing trace. The scan begins at a preset value, c (yvd), below the current location of the trace, c (yf), and continues up until the trace is located or the upper limit is reached. The criterion for finding the trace is seeing 5 out of 13 points displayed. If the trace is not found, the threshold is reduced successively to 4, 3, 2, and1 point and if it is not found at this last threshold value, control is transferred to hsj, where "nps" is typed out and control goes to the typewriter listening loop. (Note that flag 6 is cleared there as part of the cla cli 7-opr instruction — hence, the motor is stopped.)

Normally, the routine finds the trace, puts its location in <u>yf</u>, clears the IO register and interrogates the microswitch (by executing iot 11) and transfers control back to the film advance subroutine proper.

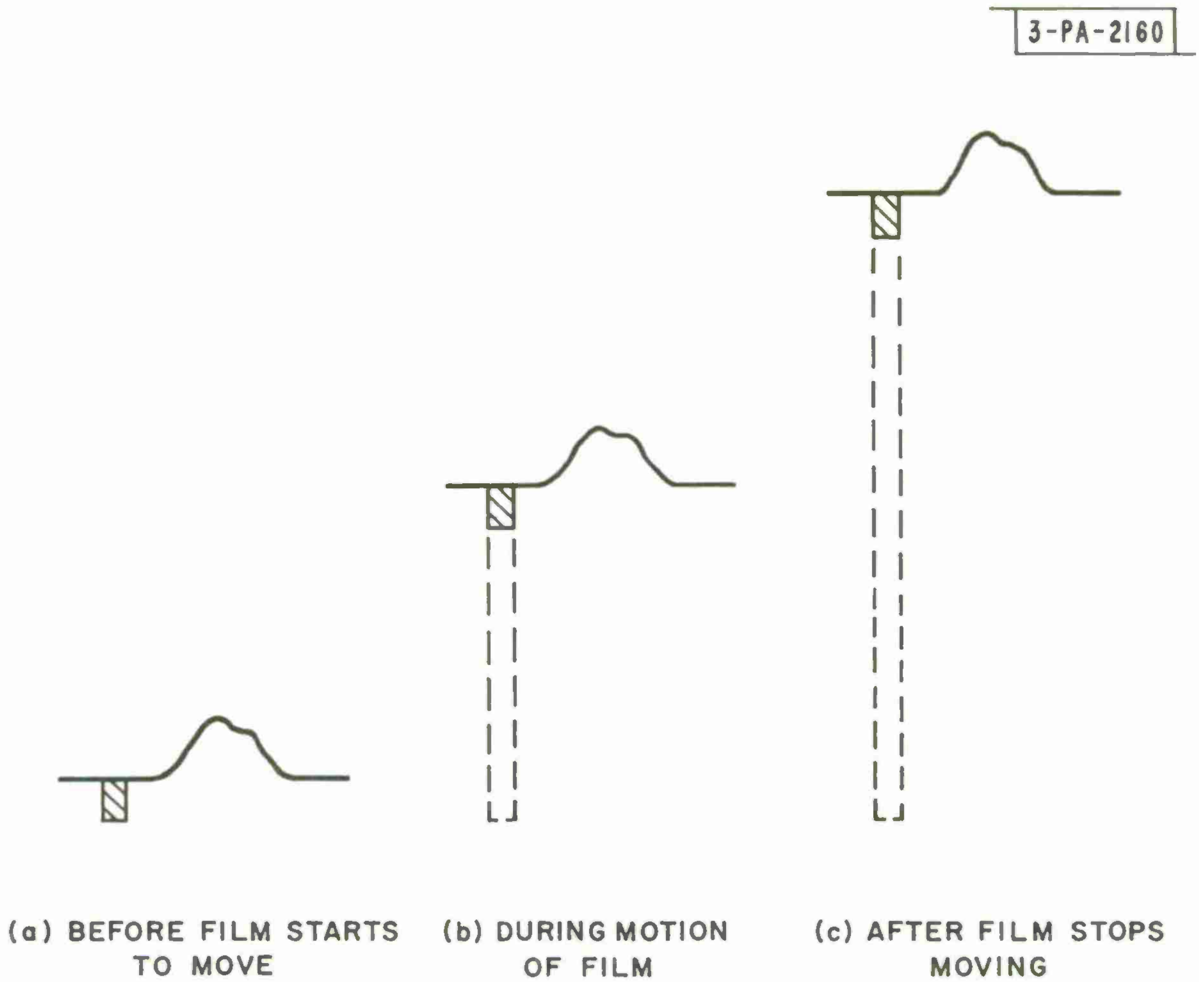
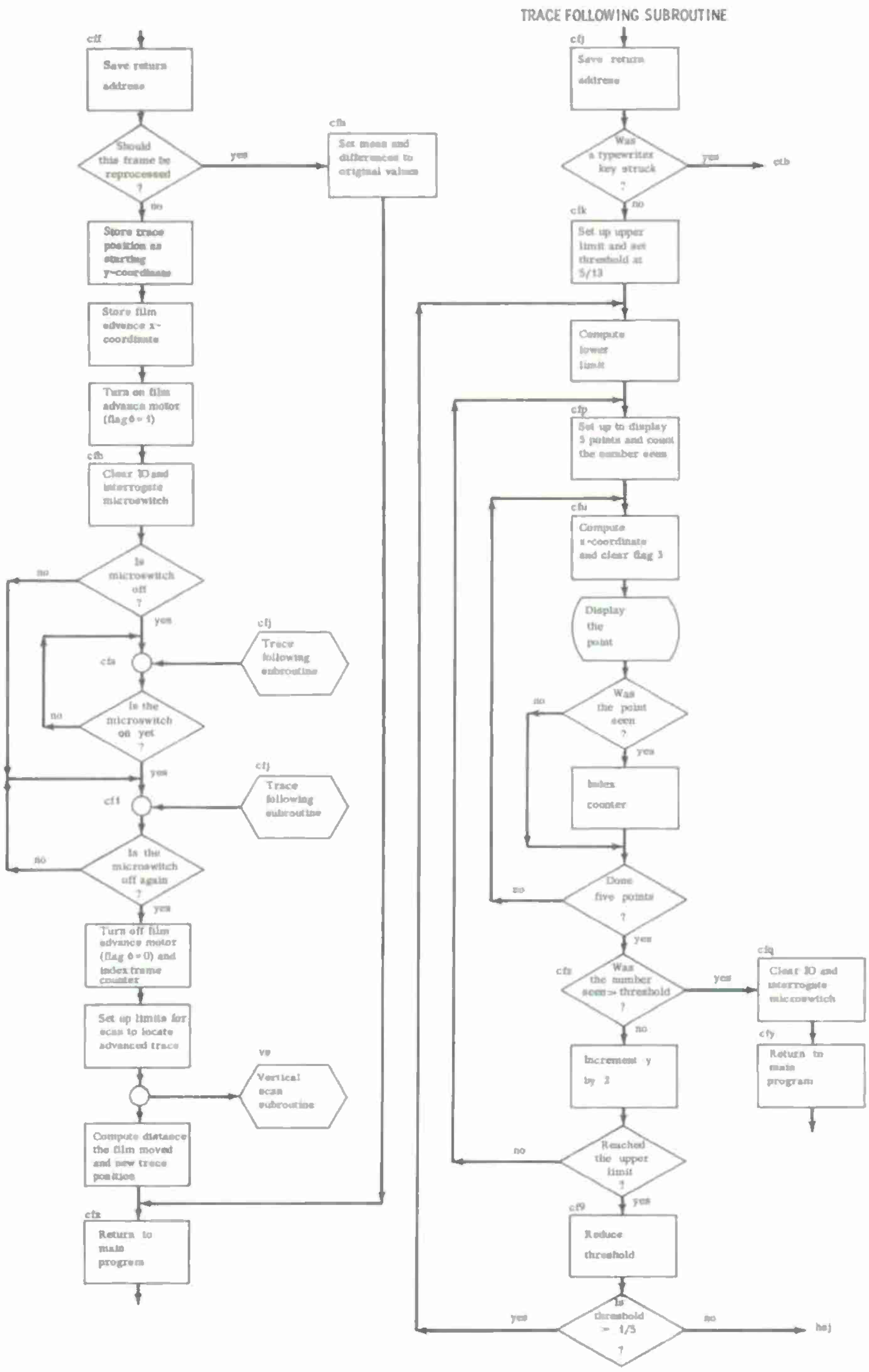


Fig. 8 Operation of trace following subroutine

FILM ADVANCE SUBROUTINE 3-PA-2176



film advance subroutine 25 feb 64

```
cff,    dap cfx             /change film frame
        szs 20
        jmp cfh
        lac ym
        dac yf              /set to follow last trace
        lac k1
        dac x
        stf 6               /motor on
cfb,    cli
        iot 11
        ril 4s
        spi i
        jmp cf1

cfa,    jsp cfj
        spi
        jmp cfa

cf1,    jsp cfj
        spi i
        jmp cf1

        clf 6               /motor off
        idx frc
        lac yf
        sar 8s
        add yvu
        dac yul
        sub yvu
        sub yvd
        dac yll
        jsp vs              /locate advanced trace
        lac yf
        sub ym
        sar 5s
        dac fav             /amount of advance
        add mm0
        dac mm0
        lac yf
        dac ym
        lac nwm
        add fav
        dac nwm
cfx,    jmp .

cfj,    dap cfy
        szf 1
        jmp ctb
        lac k2
        sal 8s
        add yf
        dac yul             /set up limits
```

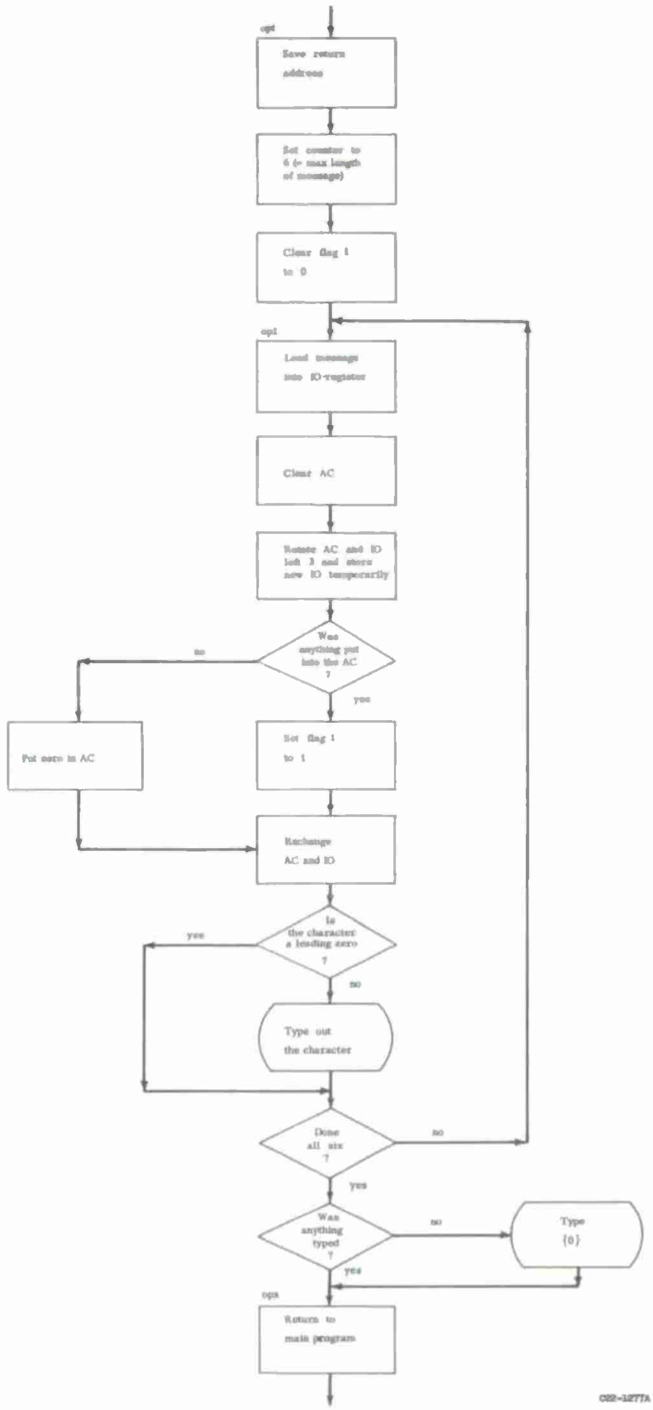
```
cfk,        init cfz, 5
            lac yvd
            sal 8s
            cma
            add yf

cfp,        swap                    /follow moving trace
            dzm t1
            setup ch, 13
cfu,        add (5
            sal 1s
            add k1
            sal 8s
            clf 3
            dpy
            szf 3
            idx t1                  /count points seen
            isp ch
            jmp cfu
cfz,        law i                   /threshold
            add t1
            sma
            jmp cfq
            swap
            add (1000
            sas yul
            jmp cfp
cf9,        law i 1
            add cfz
            dap cfz
            sas cf9
            jmp cfk+2               /try again with lower threshold
            jmp hsj

cfq,        dio yf                  /trace found
            cli
            iot 11                  /interrogate microswitch
            ril 4s
cfy,        jmp .

cfh,        lac mt2
            dac mm2
            lac mt1
            dac mm1                 /dummy frame advance
            lac mt0
            dac mm0
            add mm1
            dac nwm
            jmp cfx

start
```

9. Octal Print Subroutine

The octal print subroutine is entered by executing a jda (Jump and Deposit Accumulator) instruction with a number up to six octal digits long in the AC. This number is then typed out by the typewriter with leading zeros suppressed. If the number is negative, it is typed out in the octal equivalent of ones complement form. If the AC is clear when the routine is entered, a single zero is typed.

3-PA-2177

OCTAL PRINT SUBROUTINE



octal print subroutine 16 jan 64

```
opt,        0
            dap opx
            law i 6
            dac ch
            clf 1
op1,        lio opt
            cla
            rcl 3s
            dio opt
            sza
            stf 1
            sza i
            law 20
            rcr 9s
            rcr 9s
            szf 1
            tyo
            isp ch
            jmp op1
            szf i 1
            tyo
opx,        jmp .


start
```

10. Vertical Scan Subroutine

At various points in the program, control is transferred to the vertical scan subroutine for the purpose of locating a trace at a given x-location and between given y-limits. The most important points of reference are at hsa and hsl + 3 in the lateral scan routine, where the x-value is incremented to the left and right to take amplitude samples across the trace.

The routine begins by storing the current x-location and clearing the so-called "phase indicator" (flag 5). This flag will be set to 1 if and when a point is seen by the reader. Until the flag is set, the routine is said to be in phase 1, and after it is set, in phase 2.

The diagram in Fig. 9 is a representation of the search pattern to be described below. The series of points shown has been expanded in the x direction to make clear the order in which they would be displayed. Actually, all of the points would fall in a single vertical line.

In phase 1, the point being displayed is imaged onto the background of the film and the routine searches for the trace. It does this by displaying a series of points starting at x = c (x), y = c (yf) and continuing at x = c (x) and y = c (yf) $\pm$ 2n (n = 1, 2, 3, . .).

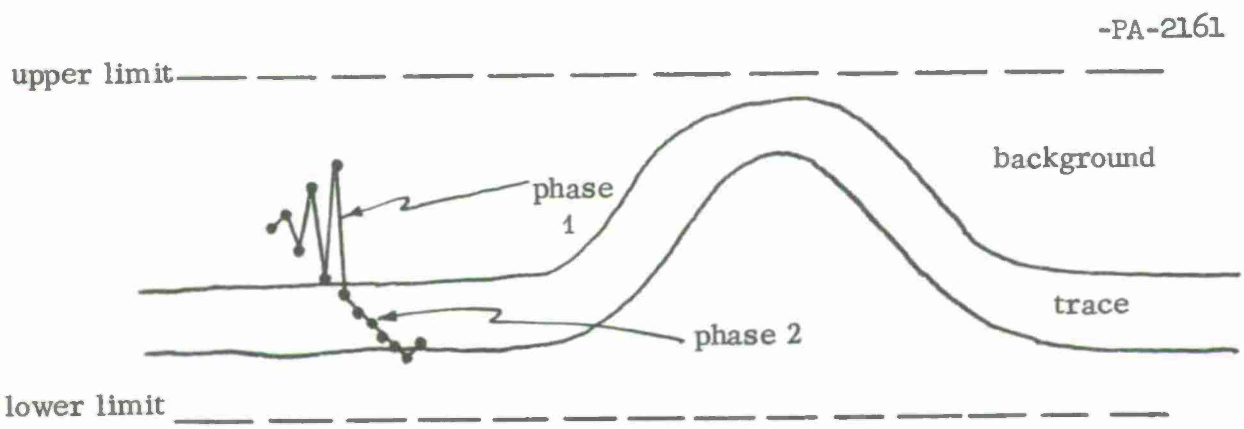


Fig. 9 Representation of search pattern

The actual sequence is c (yf), c (yf)+ 2, c (yf) - 2, c (yf)+ 4, c (yf) -4, . . . , This continues until:

(i) the upper limit is reached, in which case only the points $y = c(\underline{yf}) - 2n$ are displayed, or

(ii) the lower limit is reached, in which case only the points $y = c(\underline{yf}) + 2n$ are displayed, or

(iii) a point is seen, in which case the routine enters phase 2.

If both (i) and (ii) occur before (iii) occurs, control passes back to the main program with flag 5 = 0.

If and when phase 2 is entered, the display of points is resumed in a manner similar to phase 1. The upper and lower limits, however, are determined differently. The outputs of two "filters" are computed; one for the points $y = c(\underline{yf}) + 2n$ starting at vsr and one for the points $y = c(\underline{yf}) - 2n$ starting at vsq. The respective outputs are c (chu) and c (chd) and the functions are:

$$c(\underline{chu})_o = 400_8 \qquad\qquad c(\underline{chd})_o = 400_8$$

$$c(\underline{chu})_i = \frac{v_i + c(\underline{chu})_{i-1}}{2} \qquad\qquad c(\underline{chd})_i = \frac{v_i + c(\underline{chd})_{i-1}}{2}$$

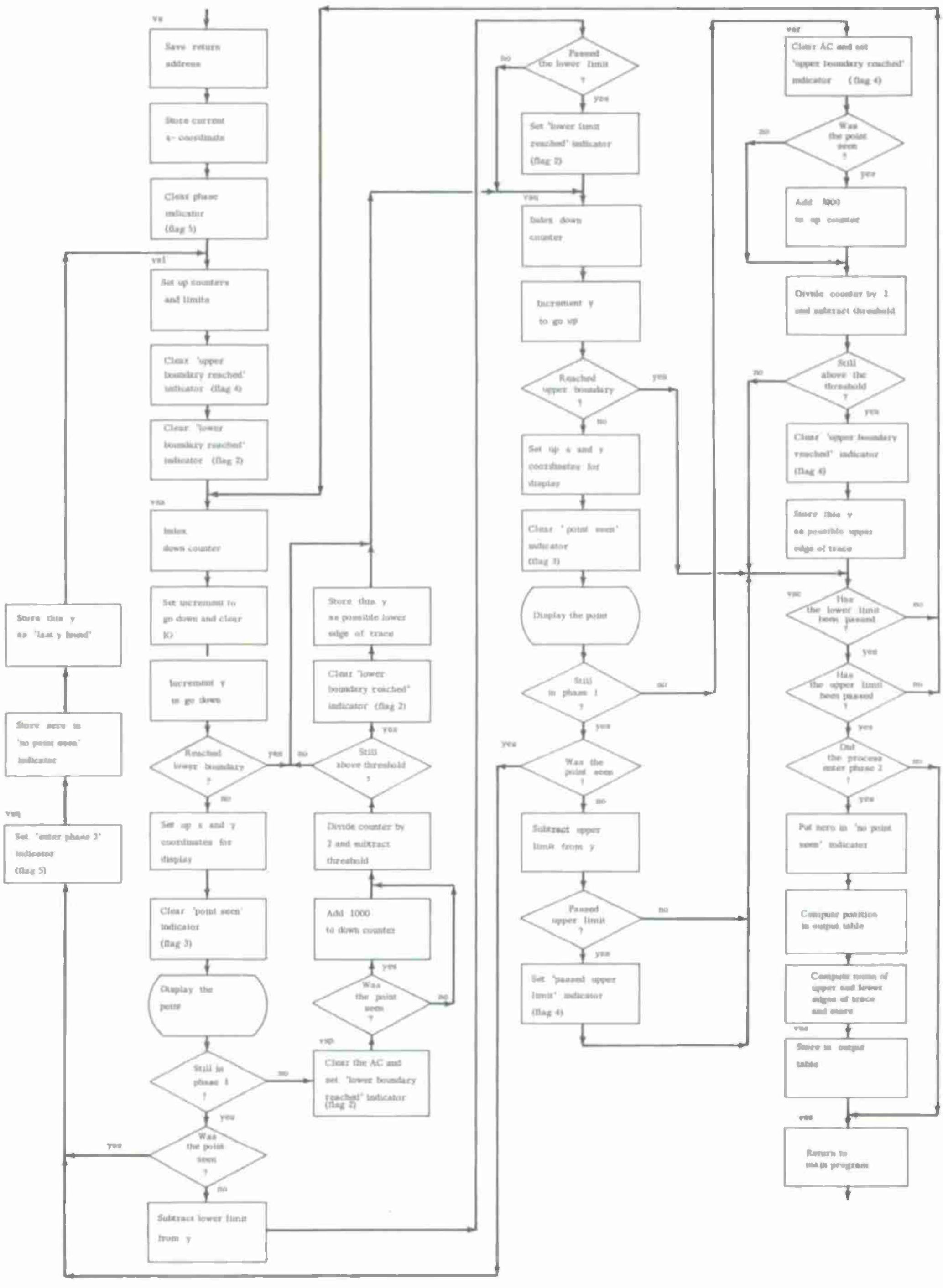
$$\text{where } v_i = \begin{cases} 1000_8 & \\ 0 \text{ otherwise.} & \end{cases}$$

As each point is stored, its y coordinate is stored as a possible value for the upper or lower edge of the trace. The edges are taken to be the values of yf last stored when the corresponding filter function falls below the threshold, c (th). (This value is currently set at 160_8.)

When both upper and lower edges of the trace have been found, the position in the output table where this sample is to be stored is computed. The value stored is the y-coordinate half-way between the upper and lower edges of the trace. Control then passes to the main program.

3-PA-2178

RADAR SCAN VERTICAL SCAN SUBROUTINE



radar scan vertical scan subroutine   7 feb 64

```
vs,         dap vsx
            lac x
            sal 8s
            dac x̄p
            clf 5                   /phase indicator

vs1,        law 400
            dac c̄hu
            dac c̄hd
            lac yf
            dac yu                  /set up limits
            sar 8s
            dac ȳd
            sar 1s
            dac y
            setup ȳdl, 1
            clf 2                   /lower boundary indicator
            clf 4                   /upper boundary indicator

vsa,        idx ydl                 /down
            cma cli-opr
            add y
            dac y
            szf 2
            jmp vsu                 /if down scan done
            lio y
            sil 9s
            lac xp
            clf 3
            dpy
            szf 5
            jmp vsp
            szf 3                   /phase 1 - find any point
            jmp vsq
            lac y
            sal 1s
            sub yll
            spa
            stf 2                   /lower boundary reached
            jmp vsu

vsq,        stf 5                   /point found, enter phase 2
            dzm nps                 /to find limits
            dio yf
            jmp vs1
```

```
vsp,        cla 12              /phase 2
            szf 3
            law 1000
            add chd
            sar 1s
            dac chd
            sub th
            spa
            jmp vsu             /below threshold, downward search done
            clf 2               /above threshold, continue
            sir 8s
            dio yd

vsu,        idx ydl             /up
            add y
            dac y
            szf 4
            jmp vsc             /if up scan done
            lio y
            sil 9s
            lac xp
            clf 3
            dpy
            szf 5
            jmp vsr
            szf 3               /phase 1 continued
            jmp vsq
            lac y
            sal 1s
            sub yul
            sma
            stf 4               /upper boundary reached
            jmp vsc

vsr,        cla 14              /phase 2
            szf 3
            law 1000
            add chu
            sar 1s
            dac chu
            sub th
            spa
            jmp vsc             /below threshold, upward search done
            clf 4               /above threshold, continue
            dio yu
```

```
vsc,        szf 2
            szf i 4
            jmp vsa             /if not done
            szf i 5
            jmp vsx             /never entered phase 2, no points seen
            dzm nps
            lac x               /record position of trace
            sub xmn
            sar s
            add (tbl
            dap vss
            lac yu
            sar 8s
            add yd
            sal 7s
            dac yf
vss,        dac .

vsx,        jmp .

start
```

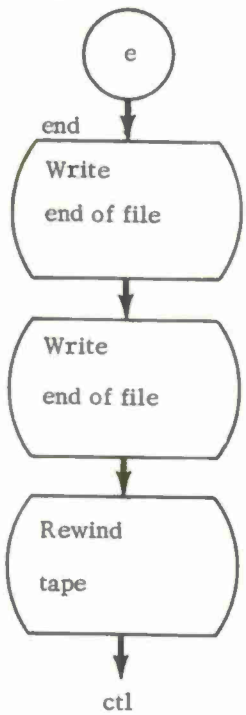
11. End Routine

When the character "e" is typed, the typewriter control routine passes control to the end routine at end, where two end-of-file marks are written and the magnetic tape is rewound. Control then returns to ctl.

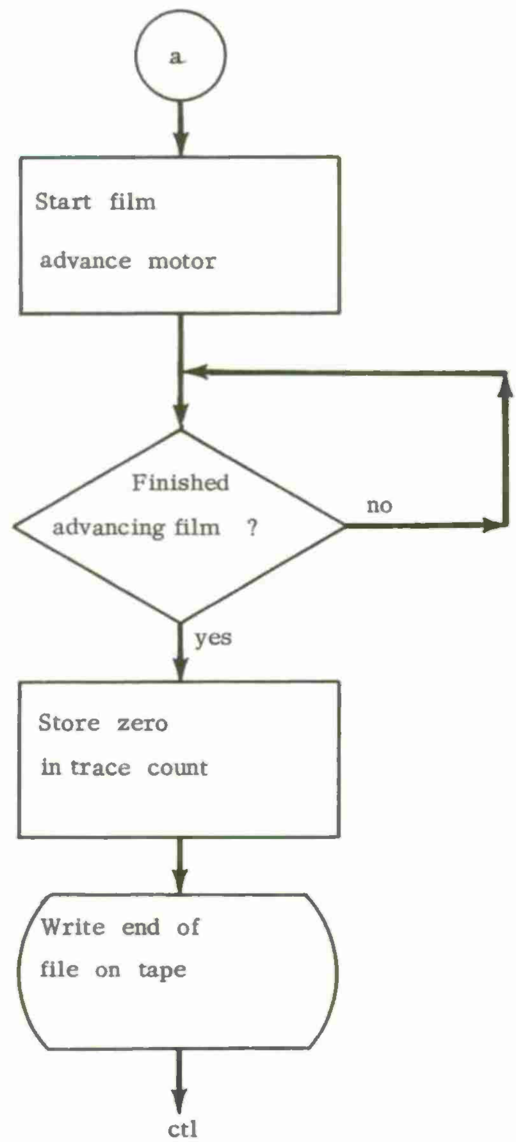
12. Advance Film Routine

The character "a" causes control to pass to adv, where the film advance motor is turned on by setting flag 6. The motor will continue to run until sense switch 6 is set to zero. Thus, if it is desired to advance the film, say between calibration sections, ss #6 is set to 1 and "a" is typed. When ss #6 is set to 0, flag 6 is cleared – the motor is turned off, the trace count is set to zero, and an end-of-file is written on the tape, and control is returned to ctl.

END ROUTINE



FILM-MOVING ROUTINE



end - move film   3 feb 64

```
/end

end,        weof
            weof
            rewind
            jmp ctl


/advance film between calibration readings

/runs motor as long as ss6 is up - stops motor, stores zero in
/trace count, writes end of file and returns to listening loop
/ when ss6 is put down


adv,        stf 6                   /turn on film advance motor
            szs 60                  /finished?
            jmp .-1                 /no
            clf 6                   /yes
            dzm pc
            weof
            jmp ctl

start
```

13. Block Output Routine

Output from the program is onto magnetic tape through the Type 51 control which requires character-by-character reading and writing. In order to get a tape in standard IBM format, use is made of a group of subroutines called the "Fletcher Routines". *
The block output routine serves to produce from a simple calling sequence:

```
law   A
jda   735
law   B,
```

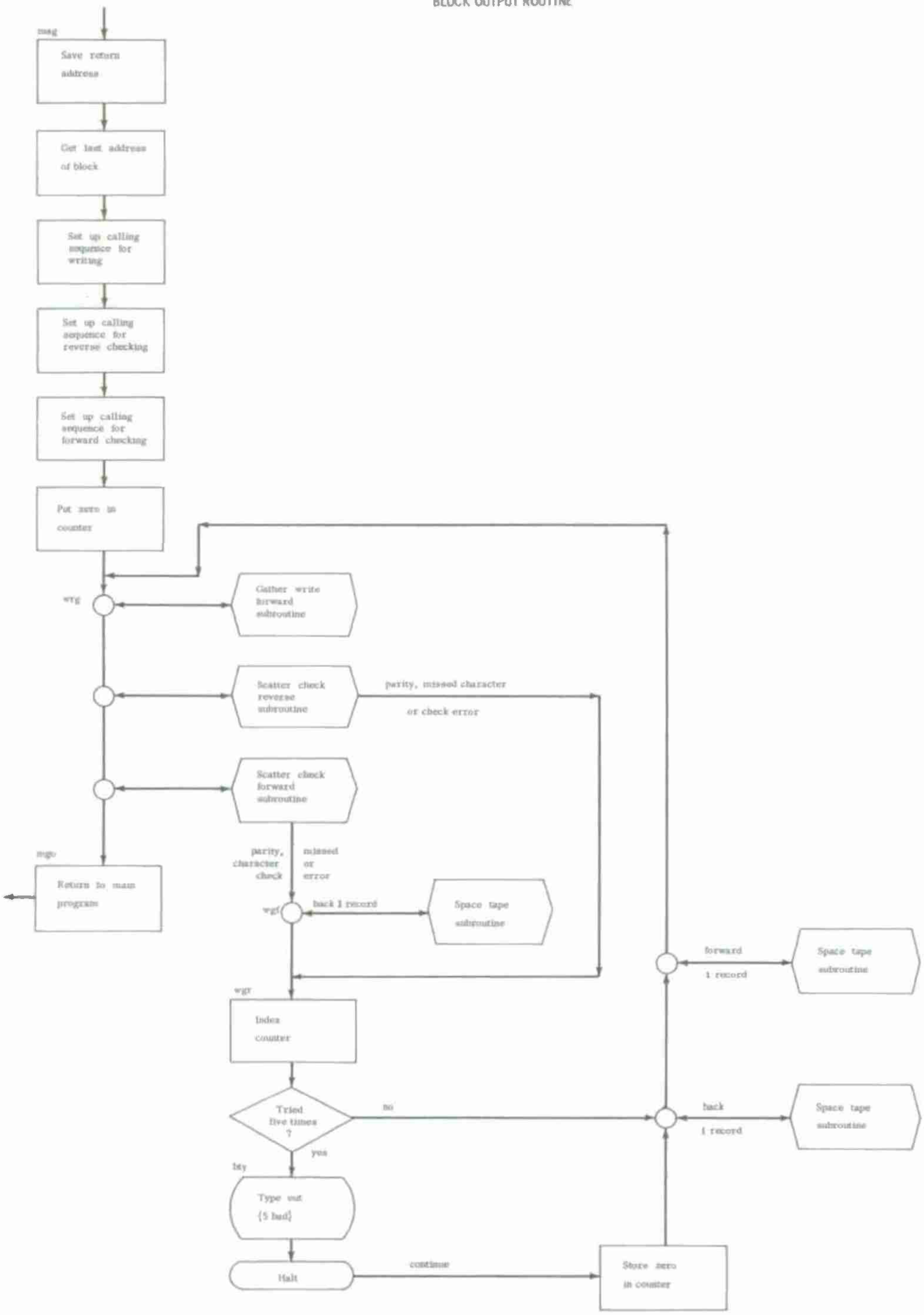
where A is the first address and B the last address of a block of registers to be written on tape, a fairly complex response, namely, to write the record on tape, check it in reverse, and check it forward. In case of error, the routine rewrites and checks reverse and forward until a total of five write-checks have been made unsuccessfully, whereupon it causes the characters "5 bad" to be typed and halts. Pressing Continue will cause another try of five write-checks.

---

* Available as DECUS No. 33 BBN-1011c Scatter Gather Mag Tape Routine in the DECAL language. Used here in a translation to MACRO, (see Appendix B).

BLOCK OUTPUT ROUTINE

3-PA-2180



block output routine   24 jan 64

```
735/

mag,        0
            dap mgo             /set up return address -1
            dap gtl             /set up to get last address
            idx mgo             /set up return address
gtl,        xct .               /get last address of block
            dac mk0             /set up calling sequences
            add (1
            dac mk1
            dac mk2
            lac mag
            dac du1
            dac du2
            sub (1
            dac du0
            dzm bdr             /put zero in counter
wrg,        jsp 51              /gwmf
du1,        0                   /first address of block
mk1,        0                   /last address +1 of block
            777777
            jsp 201             /scmr
mk0,        0                   /last address of block
du0,        0                   /first address -1 of block
            777777
            jmp wgr             /parity or miss error
            skp 600             /normal return
            jmp wgr             /check error
            jsp 220             /scmf
du2,        0                   /first address of block
mk2,        0                   /last address +1 of block
            777777
            jmp wgf             /parity or miss error
mgo,        jmp .               /normal return-return to program
            jmp wgf             /check error
wgr,        idx bdr             /index counter
            sad (5              /done 5 write-checks?
            jmp bty             /yes
            jsp 14              /no-space back one
            777776
            jsp 14              /space forward one
            1
            jmp wrg
```

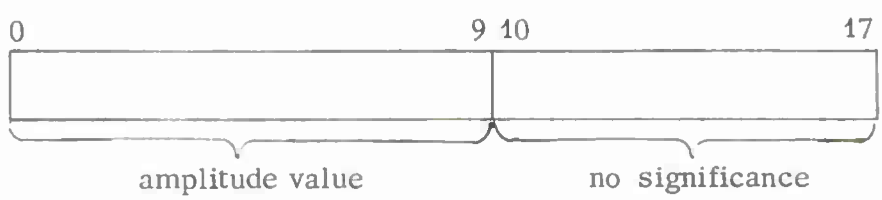
```
bty,        lio (000577         /type "5 bad"
            tyo
            rir 6s
            tyo
            rir 6s
            tyo
            lio (646162
            tyo
            rir 6s
            tyo
            rir 6s
            tyo
            hlt                 /error halt-press continue to try again
            dzm bdr             /put zero in counter
            jmp wgr 3
wgf,        jsp 14
            777776
            jmp wgr
start
```

14. Output Format

A single record is written on magnetic tape in the BCD mode, when the character "g" is typed on the typewriter. This record of 120 characters is made up of the contents of a table, tbf, set up by the titling routine. The remainder of the records on the tape are in the binary mode and are in the following format, one per trace read.

PDP Word Number	1	trace number	Control Section
	2	number of PDP words in record	
	3	number of PDP words in control section	
	4	x-coordinate of left-most sample	
	5	spare	
	6	spare	
	7	spare	
	8	checksum	
	9	data word	Data Section
	.	.	
	.	.	
	.	.	
	.	.	
	c (word 2)	data word	

Each data word has the following format:



The amplitude values are to be interpreted as ones-complement numbers. In case no point was seen, the whole word will contain the number 400000_8 (which is equivalent to -511_{10}).

constants and temporary storage   7 feb 64

```
pnd,        lac              /endcheck
mm0,        0                /medial location
mm1,        0                /first difference
mm2,        0                /second difference
nwm,        0                /new median
fmx,        6000             /max y on this frame
fht,        14000            /height of one frame...2×fmx
fav,        3640
fap,        -4000            /film advance y-coordinate
x0,         0
x,          0
y0,         0
y,          0
th,         160
xmn,        -240             /minimum x
xmx,        240              /maximum x
yll,        0                /y lower limit
yul,        0                /y upper limit
yvu,        150              /max expected ht above average
yvd,        40               /max expected ht below average
yf,         0                /last y found
k1,         0                /film advance x coordinate
k2,         400              /max expected ht for film advance


variables
constants


pc,         0                /trace counter
rsz,        0                /record size
csz,        10               /control section size
xst,        0                /x-coordinate of leftmost sample
tl1,        0                /spares
tl2,        0
tl3,        0
xyz,        0                /checksum
tba,tbl,    tba 1000/        /signal table
tbe,        tbe 1/

obl,

start beg
```

B. Series 1a

The modification to the Series 1 program to read traces of inverted orientation consists of:

(i) the insertion of one instruction into the lateral scan routine, viz., a cma between bn+5 and bn+6 – the effect of this instruction is to change the amplitude measured downward to positive and upward to negative;

(ii) in the film advance subroutine, the replacement of lac yvd by a lac k3 at cfk+2 – this has the effect of allowing the trace following routine to follow the trace closely rather than from the bottom edge of the now extensive scanning area (see Fig. 10); and

(iii) the addition of the constant k3 to the list of constants.

-PA-2162

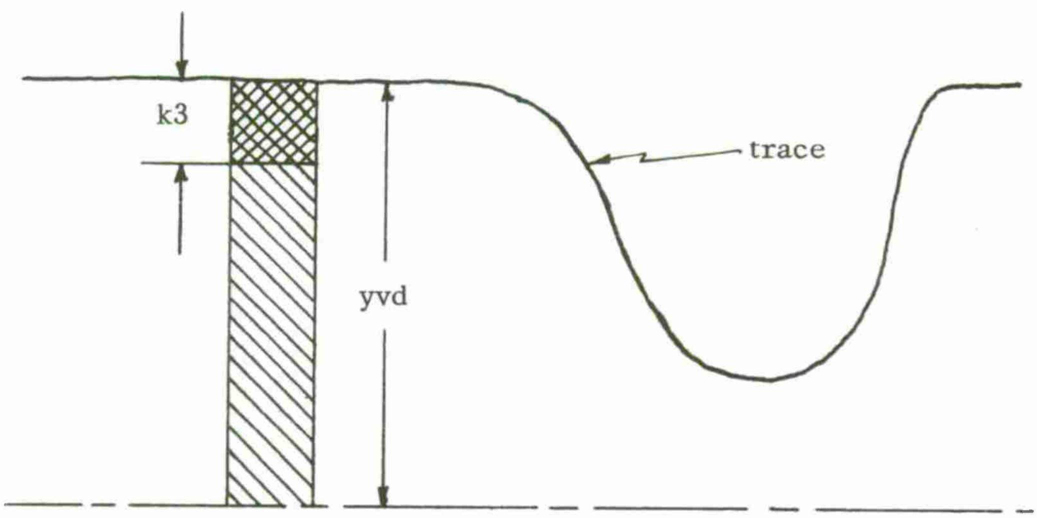


Fig. 10 Trace following

C. Series 2

The program to read film whose traces have a fiducial mark must provide, in addition to all of the routines of the earlier Series 1 program, a routine to locate the fiducial marks and must provide for reading the positions of the marks and computing the amplitudes of the traces with respect to them

The fiducial marks are located by the use of the macroscope w/reference marks routine, and their location is recorded by use of the fiducial marks location calculation routine. The latter routine is entered by typing "f" when the program is in the typewriter control loop. (The addition of the character "f" to the dispatch table is the only difference between the Series 2 typewriter control program and the earlier versions.)

1. Macroscope w/Reference Lines

As implied by its name, this routine is the earlier macroscope with an addition which generates reference lines. These lines are coordinated with the macroscope redisplay in such a way that if a point in the redisplay area is chosen, the corresponding point in the raster may be determined.

When the character "m" is typed with sense switch 5 set to zero, the routine behaves as previously described until the sixteen partial rasters have been displayed, whereupon, at ma -5, it sets up and displays a horizontal and a vertical line through the origin. These lines are displayed after every complete raster cycle of the main macroscope routine. If sense switch 5 is set to one, the program interprets the test word switches as the x-coordinate of an x= constant line and the y-coordinate of a y= constant line: x controlled by positions 0-8 and y by positions 9-17.

When, thereafter, a typewriter character is struck, the program will cause these coordinates to be typed out in the form

x m

y n

where m and n are octal 7's complement integers. These numbers are the coordinates

-PA -2163

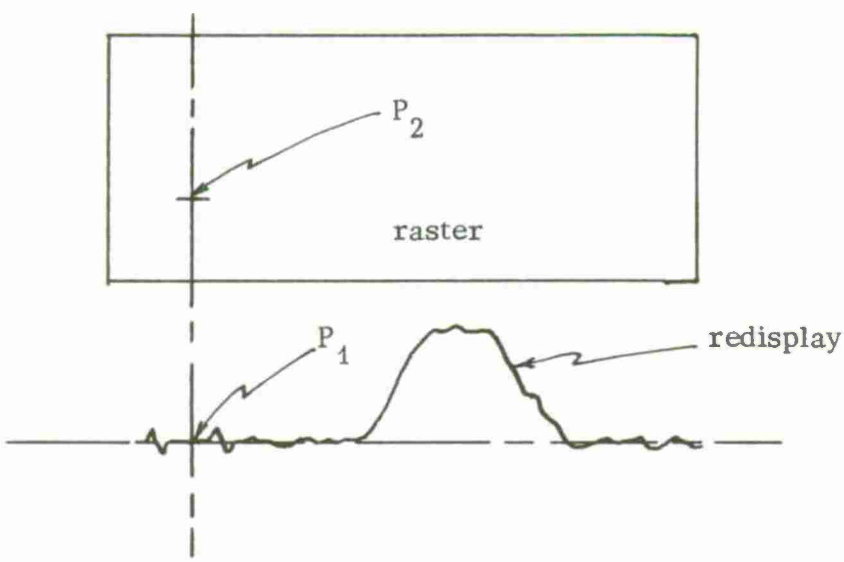
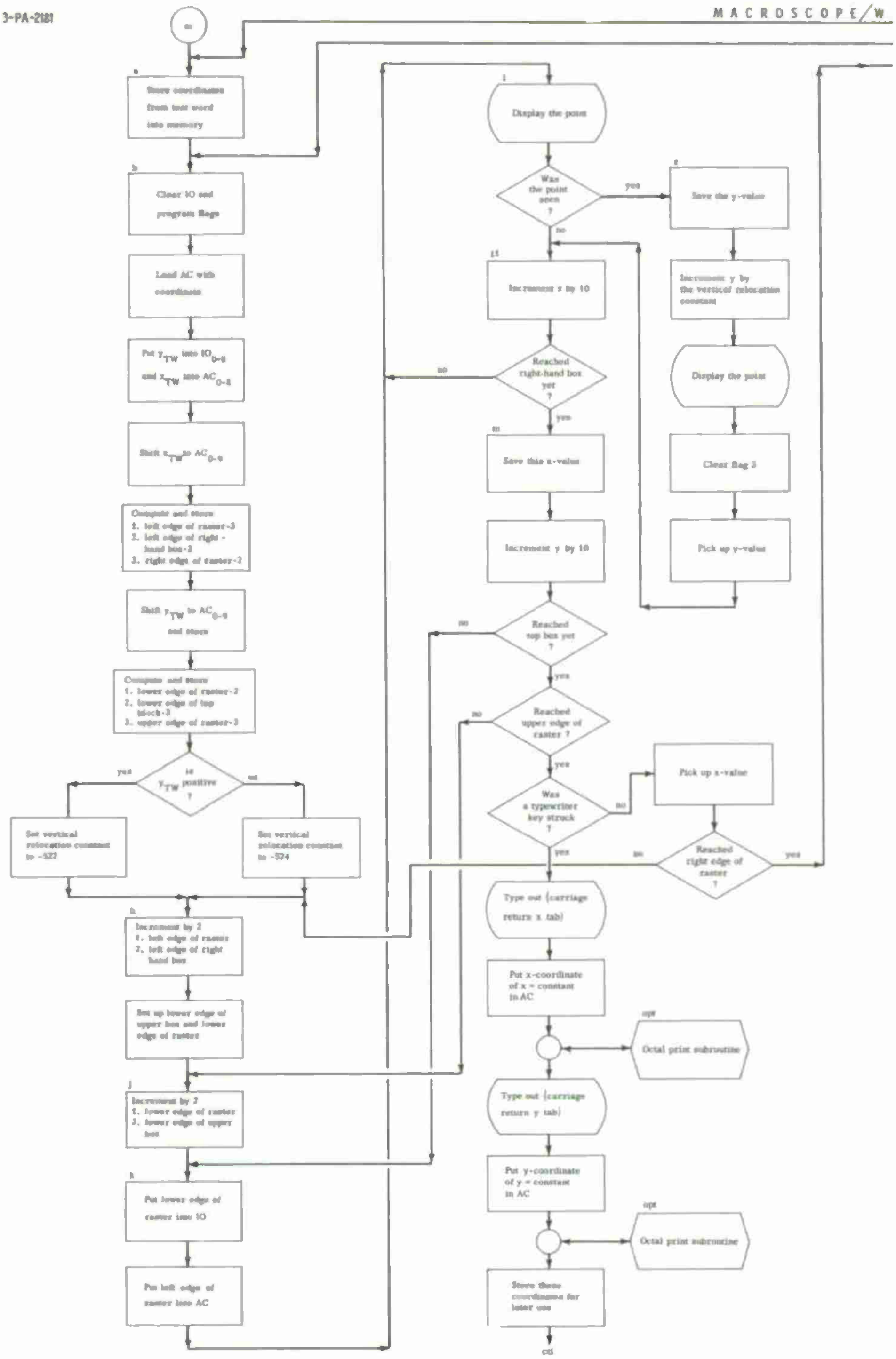


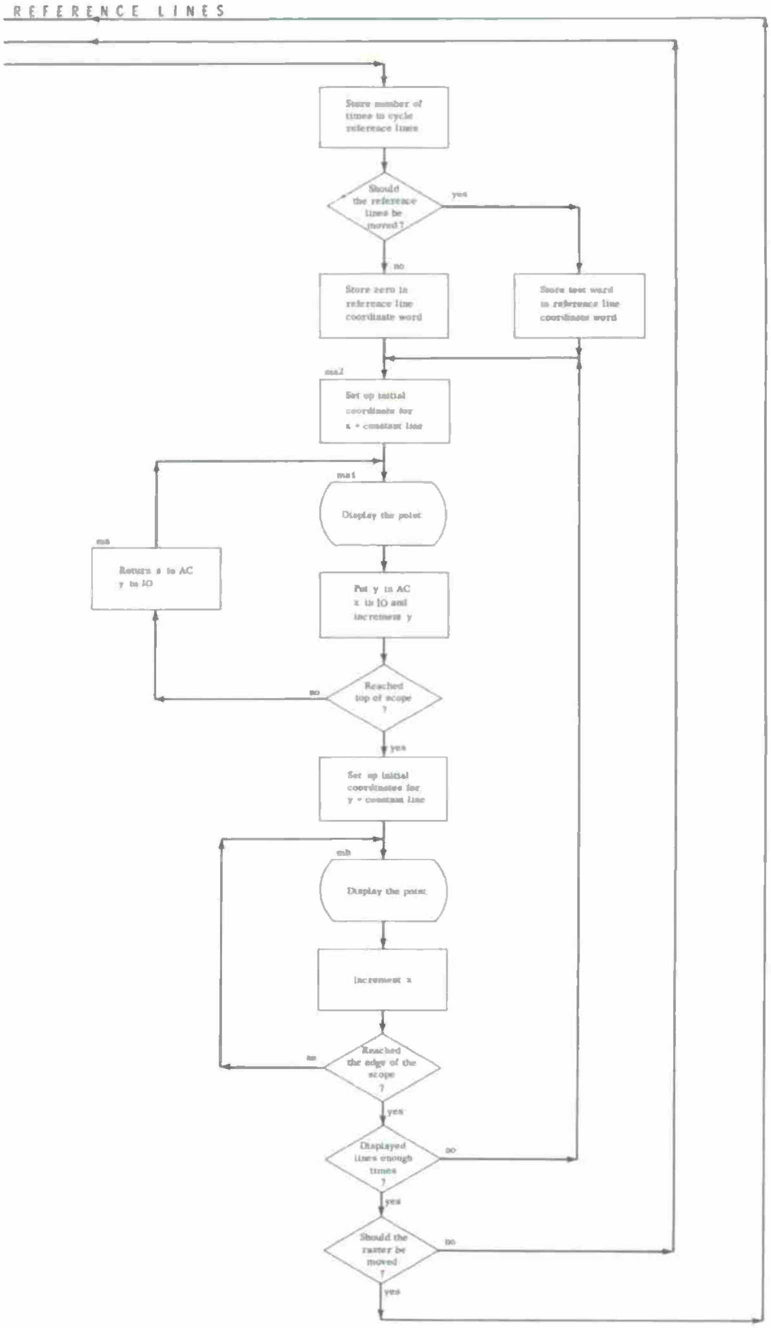
Fig. 11 Macroscope w/reference lines format

of the point in the raster corresponding to the point in the redisplay chosen. That is, if in Fig. 11, the point P_1 is the intersection of the reference lines, the coordinates of P_2, the corresponding point in the raster will be typed out. This number is also stored for use in the fiducial marks position calculation routine.

Setting sense switch 5 back to zero before typing a character returns the routine to its earlier state where the contents of the test word are interpreted as the coordinates of the raster.


3-PA-2181
MACROSCOPE/W
Store coordinates from test word into memory
Clear IO and program flags
Load AC with coordinate
Put y_{TW} into IO_{0-8} and x_{TW} into AC_{0-8}
Shift x_{TW} to AC_{0-9}
Compute and store
1. left edge of raster-3
2. left edge of right-hand box-2
3. right edge of raster-2
Shift y_{TW} to AC_{0-9} and store
Compute and store
1. lower edge of raster-2
2. lower edge of top block-3
3. upper edge of raster-3
is y_{TW} positive ?
yes
Set vertical relocation constant to -522
Set vertical relocation constant to -524
Increment by 2
1. left edge of raster
2. left edge of right hand box
Set up lower edge of upper box and lower edge of raster
Increment by 2
1. lower edge of raster
2. lower edge of upper box
Put lower edge of raster into IO
Put left edge of raster into AC
Display the point
Was the point seen ?
yes
no
Save the y-value
Increment x by 10
Increment y by the vertical relocation constant
Reached right-hand box yet ?
no
yes
Display the point
Save this x-value
Clear flag 3
Increment y by 10
Pick up y-value
Reached top box yet ?
no
yes
Reached upper edge of raster ?
no
yes
Pick up x-value
Was a typewriter key struck ?
no
yes
Reached right edge of raster ?
no
yes
Type out (carriage return x tab)
Put x-coordinate of x = constant in AC
opt
Octal print subroutine
Type out (carriage return y tab)
Put y-coordinate of y = constant in AC
opt
Octal print subroutine
Store these coordinates for later use
ctd

REFERENCE LINES



3-PA-2155

macroscope w/ reference lines 12 jan 64

```
a,          lat
            dac mxy
b,          cli 7
            lac mxy
            rcr 9s                  /y to io
            ral 9s
            sar 1s
            sub (200000+1000
            dac 1x̅0
            add (400000
            dac 1x̅1                 /set up boundaries
            add (4000
            dac 1x̅m
            swap
            sar 1s
            dac t̅
            repeat 3, sar 2s        add t
            sub (125000+1000
            dac y̅00
            add (250000
            dac y̅11
            add (4000
            dac y̅mx
            lac t
            lio (-251777
            spa
            lio (252000
            dio v̅r

h,          law 1000                /horizontal interlace
            add 1x1
            dac 1x1
            xct h
            add 1x0
            dac 1x0
            lac y11
            dac 1y̅1
            lac y00
            dac 1y̅0

j,          xct h                   /vertical interlace
            add 1y1
            dac 1y1
            xct h
            add 1y0
            dac 1y0
```

```
k,          swap
            lac 1x0

l,          dpy
            szf 3
            jmp r
l1,         add (4000           /horizontal sweep
            sas 1x1
            jmp l

m,          dac t
            swap
            add (4000           /vertical sweep
            sas 1y1
            jmp k
            sas ymx
            jmp j
            szf 1
            jmp mc
            lac t
            sas 1xm
            jmp h
            law i 2             /number of times to cycle lines
            dac t4
            cla
            szs 50              /move reference lines?
            lat                 /yes
            dac pxy             /no
ma2,        lac pxy             /set up x = constant line
            and (777000
            lio (400000
            jmp ma1
ma,         swap
ma1,        dpy
            swap
            add (1000           /increment y
            sas (400001         /reached top of scope?
            jmp ma              /no
            cli                 /yes - set up y = constant line
            lac pxy
            and (777
            rcr 9s
            lac (400000
mb,         dpy
            add (1000           /increment x
            sas (400001         /reached edge of scope?
            jmp mb              /no
            isp t4
            jmp ma2
            szs 50
            jmp b
            jmp a
```

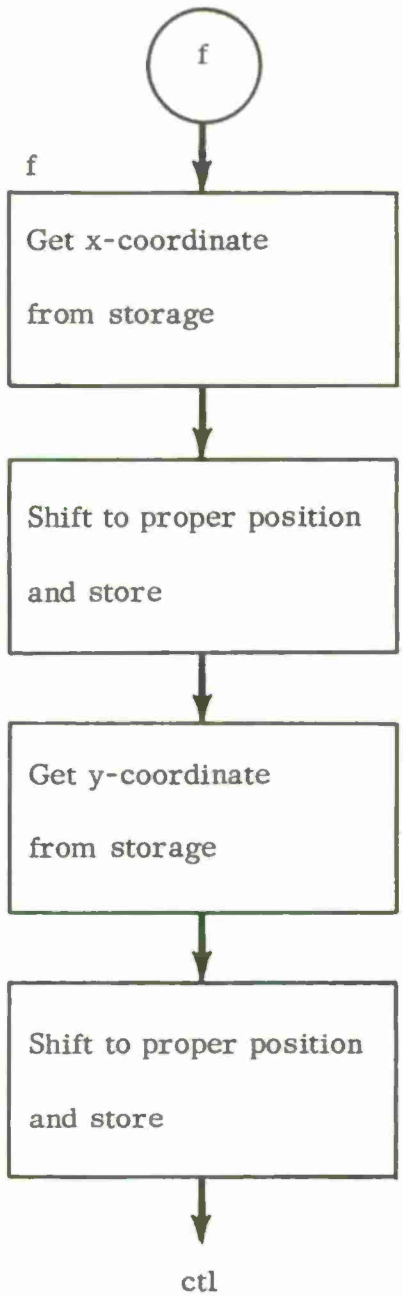
```
mc,         lio (772736
            ril 6s
            tyo                 /carriage return
            ril 6s
            tyo                 /x
            ril 6s
            tyo                 /tab
            lac pxy
            sar 9s
            dac xh1
            sal s
            jda opt             /print x-coordinate

            lio (773036
            ril 6s
            tyo                 /carriage return
            ril 6s
            tyo                 /y
            ril 6s
            tyo                 /tab
            lac pxy
            and (777
            ral 9s
            spa
            ior (777
            sub vr
            dac xh2
            sar 8s
            jda opt             /print y-coordinate
            lac xh1
            lio xh2
            rcl 9s
            dac lxy
            jmp ctl             /return to listening loop

r,          dio t               /point seen
            swap
            add vr              /vertical relocation
            swap
            dpy
            clf 3
            lio t
            jmp l1

mxy,        0                   /x and y for raster
pxy,        0                   /x and y for reference lines
lxy,        0                   /x and y for output

start
```

2. Fiducial Marks Position Calculation Routine

This short routine is entered when the character "f" is typed. It takes the stored value of the coordinates of the center of the fiducial mark, unpacks them and stores the results in the proper form for use by the lateral scan routine, the vertical scan subroutine and the film advance subroutine.

3-PA-2182

FIDUCIAL MARKS POSITION CALCULATION



fiducial marks position calculation   20 mar 64

```
f,          lac lxy               /record fiducial coordinates
            sar 9s
            sal s
            dac f1x̄
            lac lxy
            and (777
            ral 9s
            spa
            ior (377
            rar 8s
            dac f1ȳ
            jmp ctl

start
```

3. Lateral Scan Routine

For traces with fiducial marks, some provision must be made in the lateral scan routine to read the marks and store the measured amplitudes. For this purpose, a table of 75_8 registers is saved starting at register fdl by a 'dimension' pseudo-operation at the beginning of the routine. During each pass through the main loop of this routine, i.e., just before each new trace is scanned, the table is cleared by storing in each register the "no point seen" symbol, 400000. This is done by a group of instructions starting at scy -2 and extending to, but not including hs.

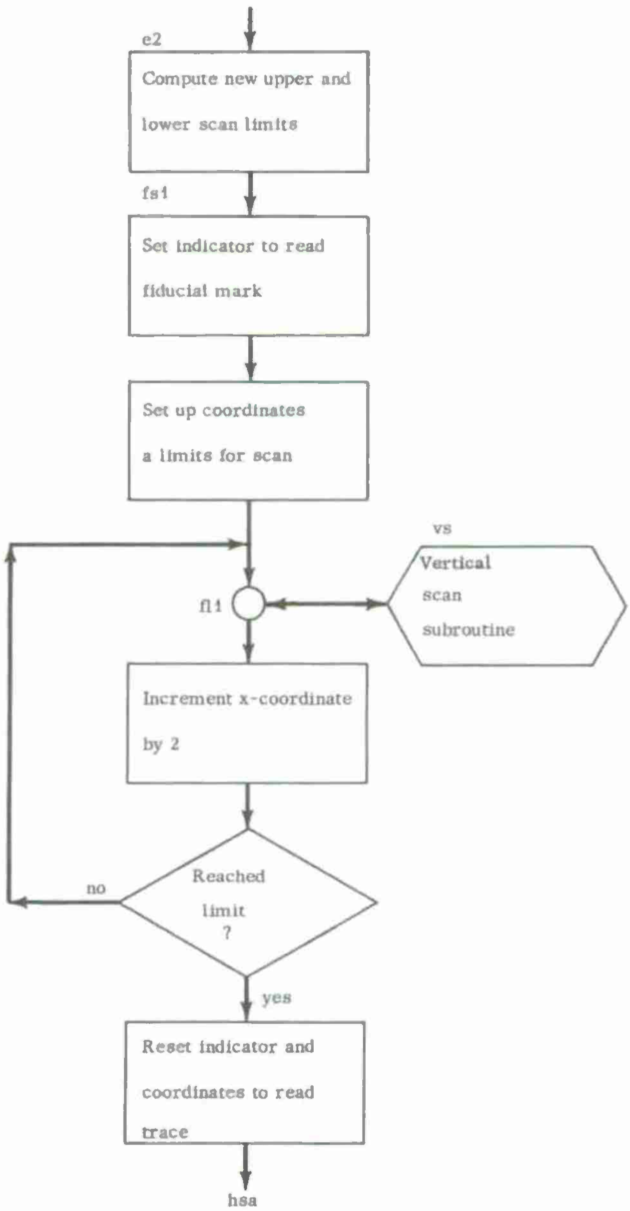
Following the seven instructions at e2 which set up the upper and lower limits for the scan, new instructions are inserted – fsl to, but not including, hsa – to read the fiducial mark. First an indicator is set to show the vertical scan subroutine that the fiducial mark is being read and then coordinates and limits are set up. The routine assumes that the x-coordinate corresponding to the center of the trace is given – c (flx) – as well as the half-width of the mark – c (dlf). The routine will scan from the (computed) left edge of the mark to the (computed) right edge. When the scanning is finished, the indicator is reset and x and y coordinates restored.

The remainder of the routine is the same as the Series 1 coding except that the value used to normalize the trace amplitudes, c (p2m), is the average of the fiducial mark amplitudes rather than of the trace amplitudes.* This requires a change of two instructions: one at hst and one at hsi. The average computed is stored also at ym for use in the film advance subroutine.

---

* That is, the amplitudes recorded by the Series 1 program are measured with respect to the average of all trace amplitudes (see p.44) while the Series 2 and 3 programs recorded amplitudes measured with respect to the average of the fiducial mark amplitudes, which is itself a measure of the noise level.

CHANGES TO LATERAL SCAN ROUTINE*



*These boxes replace box e2 in the Series 1 diagram (p. 48). C22-1357 lateral scan routine 25 mar 64

dimension fdl(75)

```
pra,        lac mm0
            add mm1
            dac nwm
            lac xmn                 /store left-most x in output table
            sal 8s
            dac xst
            lac xmx                 /compute length of output record
            sub xmn
            sar s
            add (2
            add csz
            dac rsz
            dzm sc

e1,         szf 1
            jmp ctb
            lac nwm
            sal 5s
            dac y0                  /initialize vertical scan
            dac yf
            dac ym
            sar 8s
            add yvu
            dac yul
            sub yvu
            sub yvd
            dac yll
            law tbl
            dap scu
            lio (400000
scu,        dio .                   /clear tables to nothing seen
            index scu, (dio tbe, scu
            law fdl
            dap scy
scy,        dio .
            index scy, (dio fdl 75, scy
```

```
hs,         law 1
            dac n̄ps
            lac x0
            dac x
            jsp vs                /find trace to refine position
            lac yf
            dac ym
            sar 5s
            dac p̄m
            szs 10                /should the same trace be repeated?
            jmp e2                /yes
            isp sc                /no
            jmp e
skn,        law i 1               /(-1)(number to skip + 1)
            dac sc

e2,         lac yf
            sar 8s
            add yvu
            dac yul
            sub yvu
            sub yvd
            dac yll
fs1,        law 1                 /set to read fiducial marks
            dac s̄in
            lac flx               /set up coordinates and limits
            add dlf
            dac fmn
            lac flx
            sub dlf
            dac x
            lac y0
            sal 8s
            dac yf
fl1,        jsp vs                /read mark
            law 2
            add x
            dac x
            sas fmn
            jmp fl1

            dzm sin               /set to read trace
            lac y0
            dac yf
            lac x0
            dac x

hsa,        jsp vs
            law 2
            add x                 /right
            dac x
            sas xmx
            jmp hsa
```

```
hsb,        lac x0              /reset to go left
            dac x
            lac ym
            dac yf

hsl,        law i 2             /left
            add x
            dac x
            jsp vs
            lac x
            sas xmn
            jmp hsl

hsc,        lac nps             /done
            sza                 /anything seen?
            jmp hsj             /no, complain

hst,        init hs1, fdl       /calculate average of fiducial marks
            dzm t1
            dzm p1m

hs1,        lac
            sad (400000
            jmp hsi
            sar 8s
            add p1m
            dac p1m
            idx t1

hsi,        index hs1, (lac fdl 75, hs1
            lio (flex nls
            lac t1
            sza i
            jmp erp
            lio p1m
            cla
            spi
            clc
            scl 1s
            dis t1
            hlt
            dac p2m
            sal 8s
            dac ym

rd,         szs i 60
            jmp bg
            setup t3, 12

rd9,        init rd1, tba       /redisplay signal
```

```
rd0,        lac rd1
            sub (lac tba
            sal s
            add xmn
            sal 8s
            dac t1

rd1,        lac
            sad (400000
            jmp rd2
            swap
            lac t1
            dpy-i
rd2,        idx rd1
            sas (lac tbe
            jmp rd0
            isp t3              /displayed ten times?
            jmp rd9             /no
            szs 40              /yes - hold the display?
            jmp rd              /yes

bg,         init bn, tba        /no - set up to normalize signal
bn,         lac
            sad (400000
            jmp bn1
            sar 8s
            sub p2m̄              /normalize signal
            sal 8s
            dac i bn
bn1,        index bn, (lac tbe, bn

dun,        szs 10              /should the same trace be repeated?
            jmp e1              /yes
            lac ym              /no
            sar 5s
            dac pm
            szs 30              /should the data be recorded?
            jmp e               /no

ca,         idx pc              /yes - index trace counter
            dzm xyz             /store zero in checksum register
            dzm ch
            law pc              /set up to compute checksum
            dap csm
            add rsz
            dap pnd
```

```
csm,        lac                    /get an entry
            xor ch                 /compute checksum
            dac ch
            idx csm
            sas pnd                /done the whole record?
            jmp csm                /no
            lac ch                 /yes - store checksum
            dac xyz
            law pc
            add rsz
            sub (1
            dap . 3
            law pc                 /record data on tape
            jda 735
            law
            clf 3
            szs i 50               /reading a block of n traces?
            jmp e                  /no
            lac pc                 /yes
            sub nt
            sma                    /read n traces?
            jmp ctl                /yes

e,          lac pm                 /no - estimate new median
            sub mm0
            sar 3s
            dac t2
            lac mm1
            sar 3s
            cma
            add mm1
            add t2
            dac mm1
            lac pm
            dac mm0                /constant velocity assumed
            add mm1
            dac nwm                /best guess for new median
ea,         sub fap                /frame advance point
            sma
            jmp e1
            jsp cff
            jmp ea

hsj,        clf 6
            lio (flex nps
            jmp erp

start
```

4. Film Advance Subroutine

This routine differs in only two instructions from the Series 1 routine: these move the film advance x-coordinate to the center of the fiducial mark. The instruction at cff+5 becomes lac flx and the one at cfu+2 becomes add flx.

5. Vertical Scan Subroutine

In this version of the program, the vertical scan subroutine has been changed slightly so that the amplitude measurements may be stored as either fiducial mark amplitudes in the table starting at fdl or as trace amplitudes in the table at tbl. This is done by checking, at vsc+6 and ff, an indicator, sin, set in the lateral scan routine. If it is a 1, then control passes to location vsf where the amplitudes are stored in the fiducial marks table; if it is a 0, the amplitude is stored in the signal table.

radar scan vertical scan subroutine   20 mar 64

```
/for traces with fiducial marks

vs,         dap vsx
            lac x
            sal 8s
            dac xp
            clf 5                   /phase indicator

vs1,        law 400
            dac chu
            dac chd
            lac yf
            dac yu                  /set up limits
            sar 8s
            dac yd
            sar 1s
            dac y
            setup ydl, 1
            clf 2                   /lower boundary indicator
            clf 4                   /upper boundary indicator

vsa,        idx ydl                 /down
            cma cli-opr
            add y
            dac y
            szf 2
            jmp vsu                 /if down scan done
            lio y
            sil 9s
            lac xp
            clf 3
            dpy
            szf 5
            jmp vsp
            szf 3                   /phase 1 - find any point
            jmp vsq
            lac y
            sal 1s
            sub yll
            spa
            stf 2                   /lower boundary reached
            jmp vsu

vsq,        stf 5                   /point found, enter phase 2
            dzm nps                 /to find limits
            dio yf
            jmp vs1
```

```
vsp,        cla 12              /phase 2
            szf 3
            law 1000
            add chd
            sar 1s
            dac chd
            sub th
            spa
            jmp vsu             /below threshold, downward search done
            clf 2               /above threshold, continue
            sir 8s
            dio yd

vsu,        idx ydl             /up
            add y
            dac y
            szf 4
            jmp vsc             /if up scan done
            lio y
            sil 9s
            lac xp
            clf 3
            dpy
            szf 5
            jmp vsr
            szf 3               /phase 1 continued
            jmp vsq
            lac y
            sal 1s
            sub yul
            sma
            stf 4               /upper boundary reached
            jmp vsc

vsr,        cla 14              /phase 2
            szf 3
            law 1000
            add chu
            sar 1s
            dac chu
            sub th
            spa
            jmp vsc             /below threshold, upward search done
            clf 4               /above threshold, continue
            dio yu

vsc,        szf 2
            szf i 4
            jmp vsa             /if not done
            szf i 5
            jmp vsx             /never entered phase 2, no points seen
```

```
          dzm nps
          lac sin


          lac x              /record position of trace
          sub xmn
          sar s
          add (tbl
          dap vss
vst,      lac yu
          sar 8s
          add yd
          sal 7s
          dac yf
vss,      dac .

vsx,      jmp .

vsf,      lac x
          sub flx
          add dlf
          sar s
          add (fdl
          dap vss
          jmp vst


start
```

6. Constants and Temporary Storage

The list of constants differs by the addition of dlf, defined as the fiducial mark half-width and the dropping of k1, the film advance x-coordinate.

D. Series 2a

This program bears the same relation to the Series 2 program as the Series 1a does to the Series 1. The changes described in Sec. A apply here as well.

E. Series 3

The Series 3 film reading program differs in many ways from the earlier series: it advances film frame-by-frame, it reads two fiducial marks and it reads and records a set of timing lights — all this in addition to reading the usual A-scope trace. One frame of the film is sketched as Fig. 12.

-PA-2164

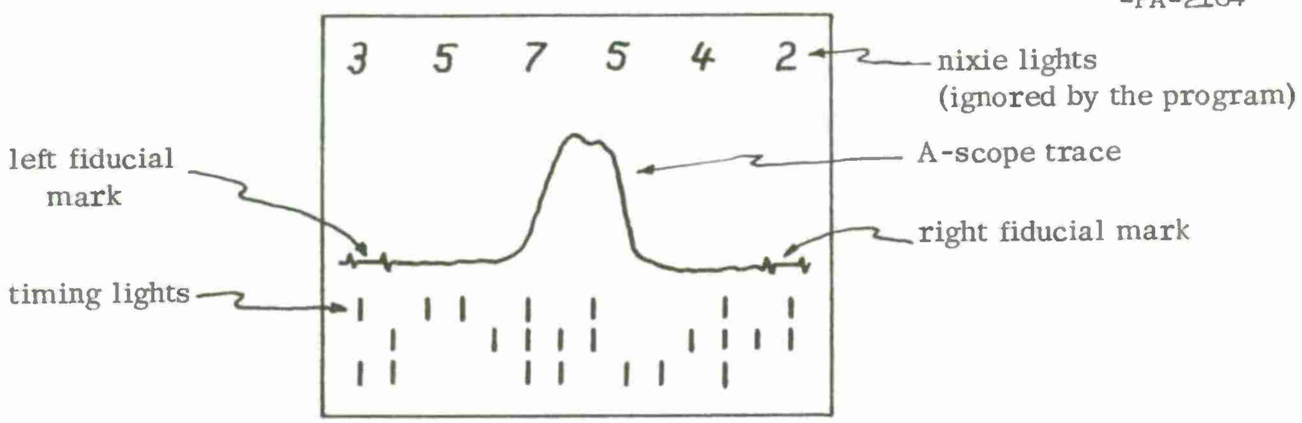


Fig. 12 One frame of project radar film

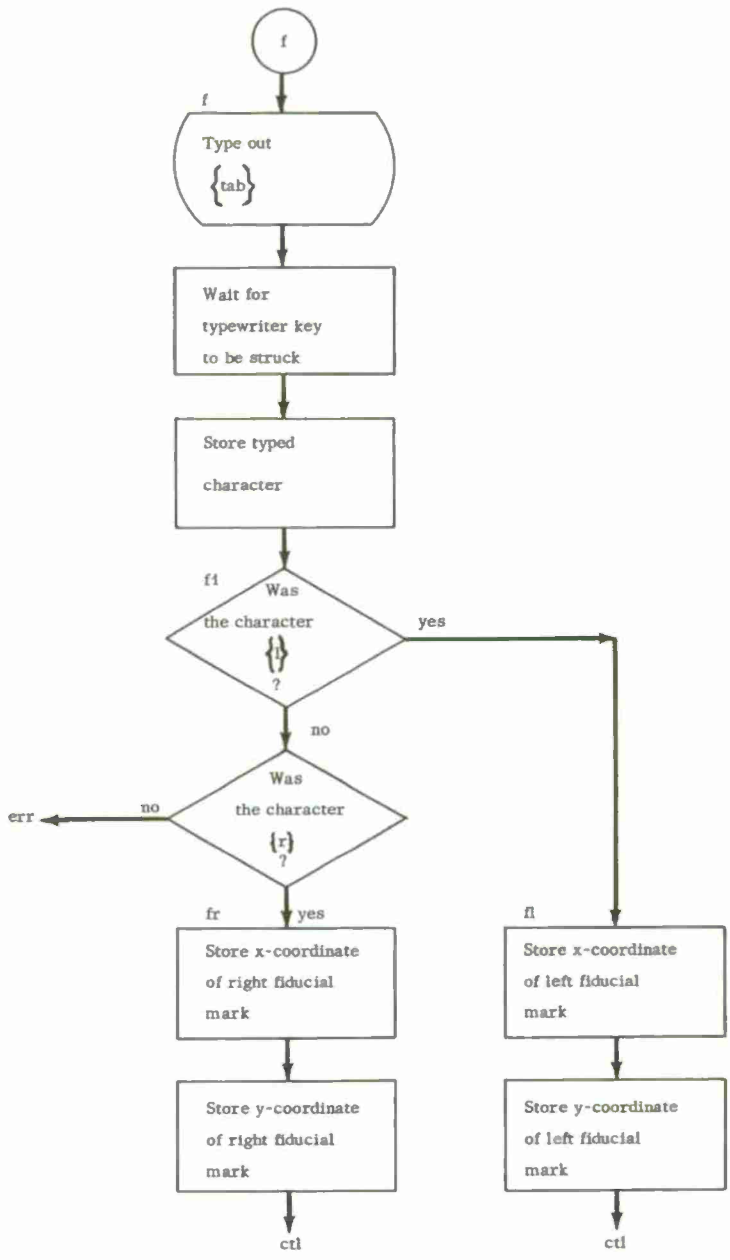
The camera which takes these pictures is rigidly mounted if front of the scope, and hence the registration of that part of the display which is fixed, i.e., the timing lights and the nixie lights, is very stable. The position of the trace on the A-scope may vary, however, from film to film, and the relative positions of the film reader and Type 30 CRT may vary from day to day.* Thus, for each film, it will be necessary to determine the locations of the trace and the timing lights as the reading is begun; in order to make this possible, a new routine, the fiducial and timing marks position calculation routine, has been added. To call this routine, the character dispatch table of the typewriter control routine has been expanded by the addition of the characters "f" and "t".

* While the Lincoln Laboratory reader is rigidly clamped so that there is a constant distance between reader and CRT, production versions of the reader with superior optics are made to allow this distance to be adjusted to suit the particular film being read.

1. Fiducial and Timing Marks Position Calculation

a. Fiducial Marks This routine is used in conjunction with the macroscope w/reference lines already described. The cycle of operation is as follows: when the reference lines have been centered on the left fiducial mark, a carriage return is typed. The computer will print out the coordinates of the point in the raster corresponding to the intersection of the reference lines. Then the character "f" is typed; the computer will respond by executing a tab. Then, when the character "l" is typed by the operator, the coordinates of the point will be properly stored for use by the lateral scan, and other routines. The same sequence is followed for the right fiducial mark, with the last control character being an "r".

FIDUCIAL MARKS POSITION CALCULATION



fiducial and timing marks position calculation   20 mar 64

```
/fiducial marks

f,          lio (36                 /execute tab
            tyo
            cla cli 7-opr
            szf i 1                 /listen for typewriter
            jmp .-1
            cla cli 7-opr
            tyi
            rcr 6s
            dac fph
            init f1, ftb
f1,         lac                     /find out if l or r
            dap fx
            xor fx
            sad fph
            jmp i fx
            index f1, (lac fe, f1
            jmp err
fx,         0
ftb,        char ll fl
            char lr fr
fe,
fl,         lac lxy                 /record left fiducial coordinates
            sar 9s
            sal s
            dac flx
            lac lxy
            and (777
            ral 9s
            spa
            ior (377
            rar 8s
            dac fly
            jmp ctl
fr,         lac lxy                 /record right fiducial coordinates
            sar 9s
            sal s
            dac frx
            lac lxy
            and (777
            ral 9s
            spa
            ior (377
            rar 8s
            dac fry
            jmp ctl
```

b. Timing Marks In order for the computer to line up properly on the timing lights, it is necessary that it be apprised of the location of the lower left light and of the x- and y- spacing between lights just before the beginning of the run. The timing marks position calculation routine was written for that purpose.

Again, use is made of the macroscope w/reference lines routine. The reference lines are placed so as to cross over the center of one of the lights, whose position in the array is known. As before, a carriage return is typed to terminate that routine. Then, assuming the lower left light is in the first column, first row, the number of the column and number of row are typed followed by the character t: thus,

XXYYt

when XX is the two-digit octal number of the column and YY is the two-digit octal number of the row. For instance,

0702t

would be for the seventh column, second row.

The computer will type out a tab and wait in a listening loop for the operator to type a character. He should type a "1". When the computer has typed a carriage return, the operator must use macroscope w/reference lines again and move the reference lines so that they cross on a timing light in a different row and a different column from the first. He then types XXYYt as before and, after the tab, the character "2". The routine will then compute the positions of the timing lights and return to the typewriter control routine.

The algorithm used may be understood by referring to Fig. 13, which shows the positions of the images of the timing lights on the scope face. The timing lights reader subroutine will display a horizontal bar across each of the 42 positions and will consider the light to have been on if two or more points are seen by the reader. The routine needs to have available the coordinates of the lower left light and the spacing between the lights. (The actual coordinates computed are those of the left most spot of the bar used to read the lower left light.)

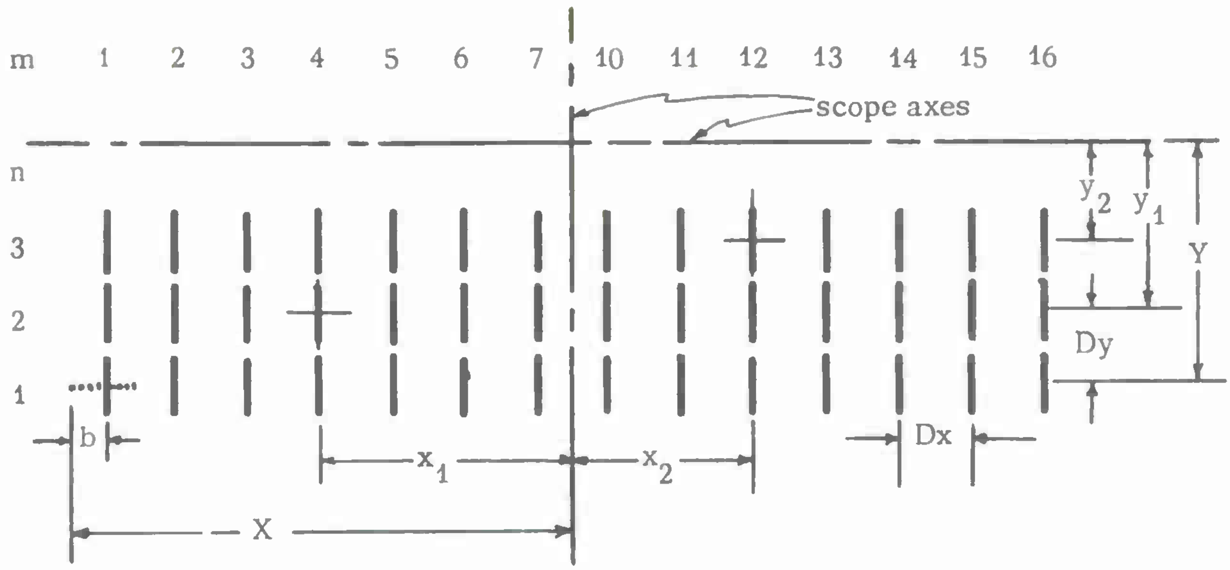


Fig. 13 Timing Lights.

The equations used are:

$$X = x_1 - (m_1 - 1)\,Dx - b$$

$$Y = y_1 - (n_1 - 1)\,Dy$$

where X, Y = coordinates of left most spot of bar used to read lower left light,

Dx, Dy = spacing between lights = $\dfrac{(x_2 - x_1)}{(m_2 - m_1)}$, $\dfrac{(y_2 - y_1)}{(n_2 - n_1)}$,

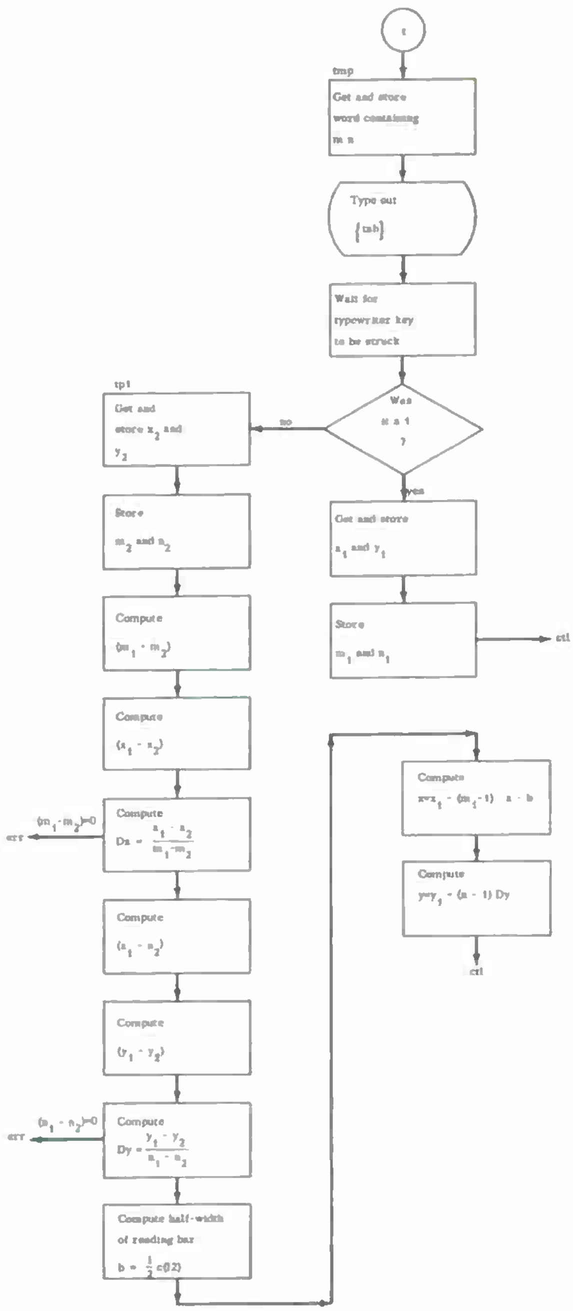
m_i, n_i = column and row numbers of timing light i, $(i = 1, 2)$,

x_i, y_i = coordinates of that light, and

b = integral part of one half of the length of the bar used to read each light = $[\frac{1}{2}\, c(12)]$.

The numbers m_1, m_2, n_1, and n_2 are supplied by the operator through the typewriter as indicated, and the values x_1, x_2, y_1, and y_2 are supplied by the macroscope w/reference lines routine (if the reference lines have been properly lined up). The number, b, is obtained from a constant stored in the timing lights reader subroutine.

TIMING MARKS POSITION CALCULATION



/timing marks

```
tmp,        lac wrd             /get mn word
            dac tmt
            lio (36             /type out tab
            tyo
            cla cli 7-opr
            szf i 1             /listen for typewriter
            jmp .-1
            cla cli 7-opr
            tyi
            rcr s
            sza i               /was the 2nd character a 1?
            jmp tp1             /no
            lac lxy             /yes - get xy word
            sar 9s
            sal s
            dac tx1             /store x1
            lac lxy
            and (777
            ral 9s
            spa
            ior (377
            rar 8s
            dac ty1             /store y1
            lac tmt
            cli
            rcr 6s
            rir 6s
            rir 6s
            dac tm1             /store m1
            dio tn1             /store n1
            jmp ctl             /return to typewriter control
tp1,        lac lxy             /get xy word
            sar 9s
            sal s
            dac tx2             /store x2
            lac lxy
            and (777
            ral 9s
            spa
            ior (377
            rar 8s
            dac ty2             /store y2
            lac tmt
            cli
            rcr 6s
            rir 6s
            rir 6s
            dac tm2             /store m2
            dio tn2             /store n2
            lac tm1             /compute (m1-m2)
            sub tm2
            dac tmt
```

```
        lac tx1                 /compute (x1-x2)
        sub tx2
        swap
        cla
        spi
        clc
        scl s
        dis tmt                 /compute Dx
        jmp err                 /(m1-m2)=0
        ral 8s
        dac ux
        lac tn1                 /compute (n1-n2)
        sub tn2
        dac tmt
        lac ty1                 /compute (y1-y2)
        sub ty2
        swap
        cla
        spi
        clc
        scl s
        dis tmt                 /compute Dy
        jmp err                 /(n1-n2)=0
        ral 8s
        dac uy
        lac l2                  /compute b
        sar s
        dac t1
        lac tm1                 /compute X
        sub (1
        mus ux
        rcl 9s
        rcl 8s
        cma
        add tx1
        sub t1
        ral 8s
        dac xnb                 /store X
        lac ty1
        ral 8s
        dac ty1
        lac tn1                 /compute Y
        sub (1
        mus uy
        rcl 9s
        rcl 8s
        cma
        add ty1
        dac yb                  /store Y
        jmp ctl

start
```

2. Lateral Scan Routine

There are two major differences between the lateral scan routines of the Series 1 and 3 programs: the latter has additional programming (1) to read the fiducial marks and (2) to redisplay the results of reading the timing lights.

Although there are two fiducial marks per trace to be read by this program, the logic used, from <u>fst</u> to <u>hsa</u>, is very similar to the routine in the Series 2 program already described. In this case, it is necessary to set up certain addresses depending on whether the left or right mark is being read, but the actual reading and the use of the indicator, <u>sin</u>, is the same. At <u>hst</u>, the readings of both marks are averaged to get the reference amplitude, from which the reported amplitudes are measured.

Earlier, at <u>hal</u>+7, immediately after the trace had been scanned, control was transferred to the timing lights reader subroutine to read the lights. Then, later on, after the trace has been redisplayed, a stretch of coding from <u>rl0</u> to <u>rd4</u> redisplays the results of reading the timing lights. The result of the reading of the lights are packed into three words in memory, as will be described below, and the coding here is merely an unpacking of these words and a display of points at the corresponding coordinate positions. The display then on the scope recreates very closely the picture on the film, except that the nixie lights and the fiducial marks are not displayed.

The remainder of the routine is the same as the earlier versions.

lateral scan routine   20 mar 64

dimension fdl(75), fdr(75)

```
pra,        lac xmn             /store left-most x in output table
            sal 8s
            dac xst
            lac xmx             /compute length of output record
            sub xmx
            sar s
            add (2
            add csz
            dac rsz
            dzm sc
            lac fly
            dac p2m

e1,         szf 1
            jmp ctb
            lac p2m
            sal 8s
            dac y0              /initialize vertical scan
            dac yf
            sar 8s
            add yvu
            dac yul
            sub yvu
            sub yvd
            dac yll

            law tbl
            dap scu
            lio (400000
scu,        dio .               /clear tables to nothing seen
            index scu, (dio tbe, scu
            law fdl
            dap scy
scy,        dio .
            index scy, (dio fdl 172, scy

hs,         law 1
            dac nps
            szs 10              /should the same trace be repeated?
            jmp fs1             /yes
            isp sc              /no
            jmp e
skn,        law i 1             /(-1)(number to skip + 1)
            dac sc
```

```
fs1,        law 1                      /set to read fiducial marks
            dac sin
fsl,        law dlf                    /set for left mark
            dap vsg
            law (fdl
            dap vsh
            law flx
            dap vsj
            lac flx                    /set up coordinates and limits
            add dlf
            dac fmn
            lac flx
            sub dlf
            dac x
            lac fly
            sal 8s
            dac yf
fl1,        jsp vs                     /read left mark
            law 2
            add x
            dac x
            sas fmn
            jmp fl1
            law drf                    /set for right mark
            dap vsg
            law (fdr
            dap vsh
            law frx
            dap vsj
            lac frx                    /set up coordinates and limits
            add drf
            dac fmn
            lac frx
            sub drf
            dac x
            lac fry
            sal 8s
            dac yf
fr1,        jsp vs                     /read right mark
            law 2
            add x
            dac x
            sas fmn
            jmp fr1

            dzm sin                    /set to read trace
            lac y0
            dac yf
            lac x0
            dac x
```

```
hsa,        jsp vs
            law 2
            add x                   /right
            dac x
            sas xmx
            jmp hsa

hsb,        lac x0                  /reset to go left
            dac x
            lac y0
            dac yf

hsl,        law i 2                 /left
            add x
            dac x
            jsp vs
            lac x
            sas xmn
            jmp hsl
            jsp q1                  /go read the timing lights

hsc,        lac nps                 /done
            sza                     /anything seen?
            jmp hsj                 /no, complain

hst,        init hs1, fdl           /calculate average of fiducial marks
            dzm t1
            dzm p1m

hs1,        lac
            sad (400000
            jmp hs1
            sar 8s
            add p1m
            dac p1m
            idx t1

hs1,        index hs1, (lac fdl 172, hs1
            lio (flex nls
            lac t1
            sza i
            jmp erp
            lio p1m
            cla
            spi
            clc
            scl 1s
            dis t1
            hlt
            dac p2m
```

```
rd,     szs i 60
        jmp bg
        setup t3, 12

rd9,    init rd1, tba           /redisplay signal

rd0,    lac rd1
        sub (lac tba
        sal s
        add xmn
        sal 8s
        dac t1

rd1,    lac
        sad (400000
        jmp rd2
        swap
        lac t1
        dpy-i
rd2,    idx rd1
        sas (lac tbe
        jmp rd0

rl0,    lac xnb                 /set up to redisplay timing lights
        add (2000
        dac lux
        dac lax
        dac lex
        lac yb
        sub (2000
        dac luy
        dac lay
        dac ley
        law i 16
        dac g1
        law i 3
        dac g2
        law i 5
        dac g3
        law tl1 2
        dap rl1
rl1,    lac                     /get a timing light word
        ral 3s
rlb,    ral s
        dac lsv
        sma                     /was a light seen?
        jmp rl3                 /no
```

```
rl2,    lac lex         /yes - set up coordinates
        lio ley
        dpy-i           /display the point
        dac lex
        swap            /increment coordinates
        add (2000
        dac ley
        isp g3          /displayed 5 points?
        jmp rl2         /no
rl3,    isp g1          /yes - done all lights in row?
        jmp rb1         /no
        isp g2          /yes - done all 3 rows?
        jmp rb2         /no
        jmp rd4         /yes
rb1,    lac lax         /set up to display next light
        add ux
        dac lax
        dac lex
        lac lay
        dac ley
        law i 5
        dac g3
        lac lsv
        jmp rlb

rb2,    lac rl1         /set up to read next word
        sub (1          /and display next row
        dap rl1
        lac lux
        dac lax
        dac lex
        lac luy
        add uy
        dac luy
        dac lay
        dac ley
        law i 5
        dac g3
        law i 16
        dac g1
        jmp rl1

rd4,    isp t3          /displayed ten times?
        jmp rd9         /no
        szs 40          /yes - hold the display?
        jmp rd          /yes

bg,     init bn, tba    /no - set up to normalize signal
```

```
bn,         lac
            sad (400000
            jmp bn1
            sar 8s
            sub p2m              /normalize signal
            sal 8s
            dac i bn
bn1,        index bn, (lac tbe, bn

dun,        szs 10               /should the same trace be repeated?
            jmp e1               /yes
            szs 30               /no - should the data be recorded?
            jmp e                /no

ca,         idx pc               /yes - index trace counter
            dzm xyz              /store zero in checksum register
            dzm ch
            law pc               /set up to compute checksum
            dap csm
            add rsz
            dap pnd
csm,        lac                  /get an entry
            xor ch               /compute checksum
            idx csm
            sas pnd              /done the whole record?
            jmp csm              /no
            lac ch               /yes - store checksum
            dac xyz
            law pc
            add rsz
            sub (1
            dap . 3
            law pc               /record data on tape
            jda 735
            law
            clf 3
            szs i 50             /reading a block of n traces?
            jmp e                /no
            lac pc               /yes
            sub nt
            sma                  /read n traces?
            jmp ctl              /yes

e,          jsp cff              /no - go advance the film
            jmp e1

hsj,        clf 6
            lio (flex nps
            jmp erp

start
```

3. Film Advance Subroutine

Since the project radar film is framed, the position of the image of the trace on the scope will always be the same. Therefore, the film advance subroutine is very simple. At cfb, the motor is turned on by setting flag 6; the microswitch is then interrogated as before until the motor has made one revolution, and when it has, the motor is turned off.

The routine then enters a delay loop for about 60 milliseconds to allow for the decay of transients in the control lines. The motor is then turned on again until it has made one more revolution, so that the film will have been advanced a total of four sprocket holes, or one frame.

Note that there is no provision here for a dummy advance; since one trace is read per film advance, repeating the same trace is equivalent to reading all traces in view. Hence, sense switch 2 is not used in this version of the program.

film advance subroutine 8 jan 64

/for framed film

```
cff,        dap cfx                 /save return address
            law i 2                 /set up to run motor twice
            dac ch
cfb,        stf 6                   /turn on motor
            cli
            iot 11
            ril 4s
            spi i
            jmp cf1
cfa,        jsp cfj
            spi
            jmp cfa
cf1,        jsp cfj
            spi i
            jmp cf1
            clf 6                   /turn off motor
            law i 7777
            dac ch1                 /set up to delay
            isp ch1
            jmp .-1
            isp ch
            jmp cfb
cfx,        jmp .

cfj,        dap cfy
            szf 1
            jmp ctb
            cli
            iot 11
            ril 4s
cfy,        jmp .

start
```

4. Timing Marks Reader Subroutine

This subroutine uses information supplied by the timing marks location calculation routine, namely, the values X = c(xnb), Y = c(yb), Dx = c(ux) and Dy = c(uy). It then proceeds to display bars of points (n points long, where n = c(12), initially 7; this value may, of course, be changed by the operator). Each bar intersects a timing light position and if the light is lit, two or more points will be seen by the reader.* The bars are displayed from bottom to top, left to right starting at the lower left corner.

When a timing light has been seen, the routine computes, from the display coordinates used, the corresponding bit in a three register table in the output block. The right-hand 14 bits of these three registers may be taken as a bit picture of the timing lights, e.g., the pattern

| | | ‡ ‡ | | ‡ ‡ | ‡ ‡ ‡ |
‡ ‡ ‡ | | ‡ ‡ | ‡ | ‡ ‡ | |
‡ | ‡ | ‡ ‡ ‡ ‡ | ‡ ‡ | ‡ ‡

on the film, where | represents light on and ‡ represents light off, gives

000 011 100 110 010 001	or	0 3 4 6 2 1
000 000 011 001 010 011		0 0 3 1 2 3
000 001 010 000 100 100		0 1 2 0 4 4

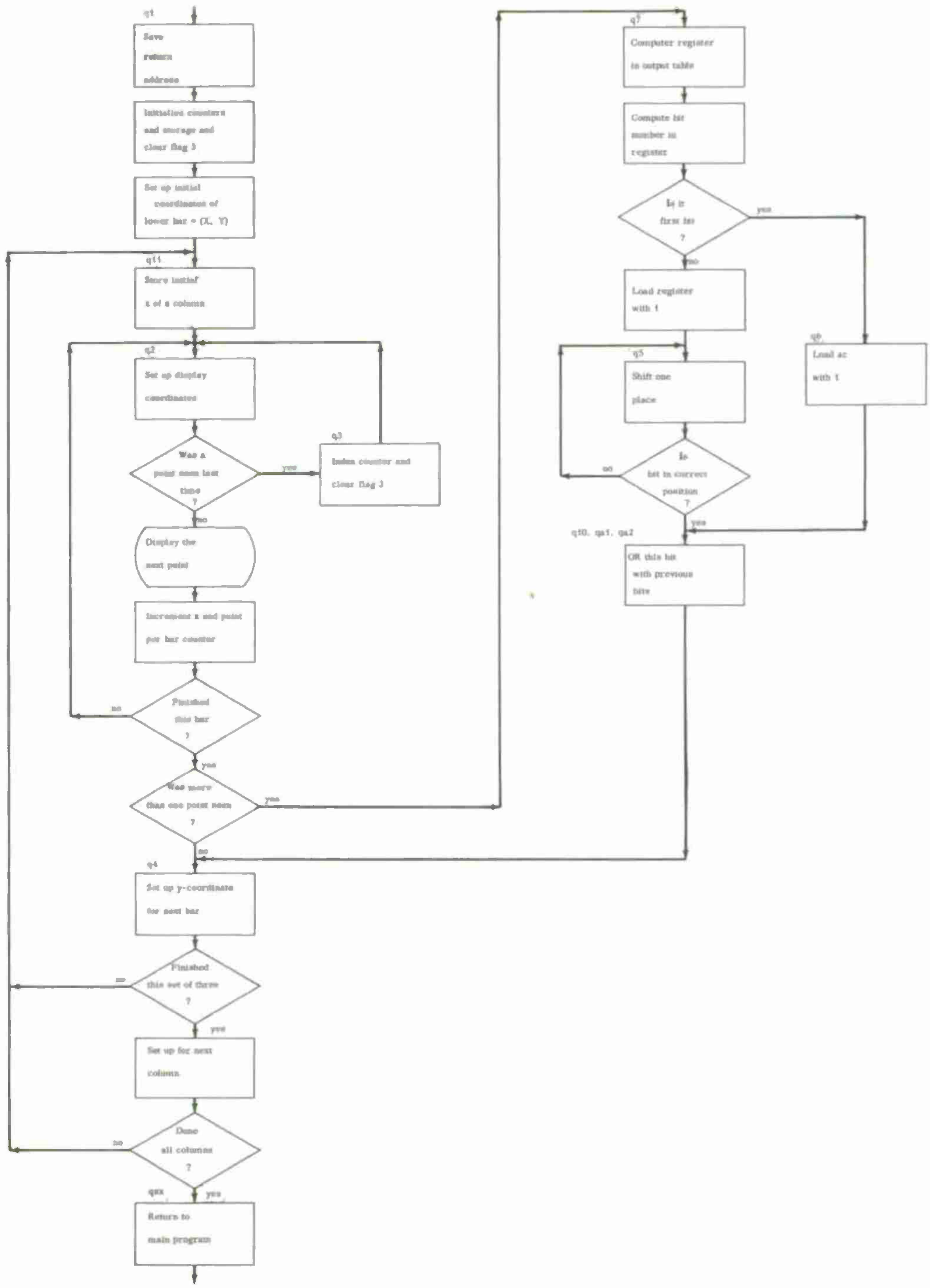
in core. These three words are written out on magtape as words 5, 6, and 7 of the control section of the output record.

---

* The bar is made up of a series of points displayed at every second addressable location on the scope. The spacing may be reduced to increase detection probability by changing the contents of dx from 1000 to 400.

3-PA-2186

TIMING MARKS READFR SUBROUTINE



timing marks reader subroutine   4 feb 64

```
q1,         dap qsx             /save return address
            dzm tk1
            dzm tk2
            dzm tl1
            dzm tl2
            dzm tl3
            dzm hb0
            dzm hb1
            dzm hb2
            clf 3
            lac yb
            dac y
            lac xnb             /set up initial x-coordinate
            dac xb              /of lower left bar = X
q11,        lac xb              /initiate x
            dac x
q2,         lac x
            lio y
            szf 3               /was a point seen last time?
            jmp q3              /yes
            dpy-i               /no - display next point
            lac x               /increment x
            add dx
            dac x
            idx tk1
            sas 12              /finished this bar?
            jmp q2              /no
            dzm tk1             /yes - zero counter
            law 1
            sub tk2
            dzm tk2
            spa                 /was more than 1 point seen?
            jmp q7              /yes - store bit in tl table
q4,         lac y               /no - go on to next bar
            add uy
            dac y
            idx hb1
            sas (3              /through with this set of 3?
            jmp q11             /no
            lac yb              /yes
            dac y
            dzm hb1
            lac xb              /move to next column
            add ux
            dac xb
            idx hb2
            sas (16             /done all columns?
            jmp q11             /no
qsx,        jmp .               /yes - return to main program
```

```
q3,        idx tk2               /count number of points seen
           clf 3
           jmp q2

q7,        lac (2
           sub hb1
           add (tl1              /compute register in tl table
           dap qa1
           dap qa2
           lac hb2
           sub (15               /compute bit number in register
           sza i
           jmp q6
           dac hb0
           law 1
           dac hb3
q5,        lac hb3               /shift bit to correct position
           sal s
           dac hb3
           isp hb0
           jmp q5
q10,       lac hb3               /OR the bit into the tl table
qa1,       ior
qa2,       dac
           jmp q4

q6,        law 1
           dac hb3
           jmp q10

xb,        0                     /x-coor of current column
yb,        0                     /y-coor of lowest row
xnb,       0                     /initial x of lowest left bar
dx,        1000                  /distance between points on bar
ux,        0                     /initial x-dist between bars
uy,        0                     /initial y-dist between bars
l2,        7                     /number of points per bar
tk1,       0                     /cntr - points displayed per bar
tk2,       0                     /cntr - points seen per bar
hb0,       0                     /cntr - shifts
hb1,       0                     /cntr - bars per column
hb2,       0                     /cntr - number of columns
hb3,       0                     /bit holder

start
```

5. Constants and Temporary Storage

Since the film is framed and no tracking is done, all constants having to do with film advancing are dropped from the list. Also, the registers <u>tl1</u>, <u>tl2</u>, and <u>tl3</u> in the output record are now identified as containing the timing light readings. Otherwise, the list is as before.

III. MODIFICATIONS TO FILM READING PROGRAM SYSTEM

Two further changes have been made to the Film Reading Program System, its programs and equipment. First, the tape system was changed to one which provided for block transfers from memory with core memory references interleaved with main frame operation and second, the assembly program was changed from MACRO to MIDAS.

A. Tape System

A new tape system, consisting of

1) a High-Speed Channel Control - Type 19
2) a High-Speed Data Control - Type 131
3) an Automatic magnetic Tape Control - Type 510, and
4) three IBM Tape Transports - Model 729 VI,

was added to the installation replacing the Type 51 Tape Control and Type 50 Transports. This new equipment allowed recording at densities of 200, 556 and 800 characters/inch and allowed for automatic block transfers to and from memory.

The effect on the film reading programs was minimized by writing a tape package which looked to the user almost exactly like the routines already used.* As a matter of fact, the same block output routine was used to set up the calling sequences.

There is one difference between the tape packages that is worth noting: it was not practicable to include the scatter-gather feature in the new package. This has not been a constraint, however, since the film reading programs themselves have not been required to use that feature.

A listing of the routines is given in Appendix D: it is a version with absolute addresses which can be assembled without fear of symbol conflict.

B. MIDAS Assembly Program

Minor modifications have been made to the programs to adapt them for the MIDAS assembler:

---

* See Appendix B.

1) the macro definitions of the Extended MACRO System (see Appendix A) have been incorporated into the first, or definitions, tape of the program;

2) certain symbols and parameter assignments have also been incorporated into that tape (see listing on following two pages);

3) pseudo-instruction names with more than four characters have been spelled out in the listing, since MIDAS recognizes symbols up to six characters long and no longer equates such symbols as flex and flexo.

film reading program system   series 1/2   20 jul 64

```
/for MIDAS compilation

/definitions

define       bintape
             dzm 7
terminate

define       bcdtape
             law 1
             dac 7
terminate

define       rewind
             jsp 633
terminate

define       weof
             jsp 415
             nop
terminate

s=1s
dpy=730007
clo=szoVspaVsma i
xx=hlt
clc=claVcma
szm=szaVsma
spq=szm i

define       sensewitch a
             szs 8xa
terminate

define       init a,b
             law b
             dap a
terminate

define       index a,b,c
             idx a
             sas b
             jmp c
terminate

define       swap
             rcl 9s
             rcl 9s
terminate
```

```
define       listen
             claVcliVclf 1
             szf i 1
             jmp .-1
             tyi
terminate

define       load a,b
             lio (b
             dio a
terminate

define       setup a,b
             law i b
             dac a
terminate

define       count a,b
             isp a
             jmp b
terminate

define       move a,b
             lio a
             dio b
terminate

define       clear a,b/c
             init c,a
c,           dzm
             index c,(dzm b 1,c
terminate


start
```

IV. CONCLUSIONS

The film reading program system described herein has been written to digitize A-scope traces in three different formats. It is evident that additional "series" could be prepared to read film in other formats, and as a matter of fact, three such programs have been written for nonproduction film reading. Since the framework is already available, the filling out of the program for a particular job is reasonably easy.

The accuracy with which the reader records the information from the film has been the subject of a recent study,* which showed that within the limits of the quantization, there is essentially no error introduced by the film reader.

Thus, the Programmable Film Reader is a flexible and accurate tool for the reduction of large amounts of film data to computer assimilable form.

---

* See Appendix E.

APPENDIX A

EXTENDED OPS AND MACROS

```
extended ops and macros  8 november 1962

lap=cla 100
oh=iot i
clo=651600
spq=650500
szm=640500

define
          senseswitch A
          repeat 3, A=A+A
          szs A
          term

define
          initialize A, B
          law B
          dap A
          term

define
          index A, B, C
          idx A
          sas B
          jmp C
          term

define
          listen
          cla+cli+clf 1-opr-opr
          szf i 1
          jmp .-1
          tyi
          term

define
          swap
          rcl 9s
          rcl 9s
          term

define
          load A, B
          lio (B
          dio A
          term
```

```
define
          setup A, B
          law 1 B
          dac A
          term

define
          count A, B
          isp A
          jmp B
          term

define
          move A, B
          lio A
          dio B
          term

define
          clear A, B
          init .+2, A
          dzm
          index .-1, (dzm B+1, .-1
          term

start
```

APPENDIX B

MAGNETIC TAPE ROUTINES FOR PROGRAMMED TAPE SYSTEM

The tape equipment used by the computer for which these film-reading programs were written requires all timing to be done under program control and requires character by character transfer between computer and tape. A set of subroutines is used to control timing and information transfer in such a way as to produce standard IBM formatted tapes. These routines were written by William Fletcher of Bolt, Beranek and Newman in the DECAL language and are now available from the Digital Equipment Computer Users Society (DECUS). The listing which follows is an instruction by instruction translation of the routines from DECAL to MACRO.

There are four routines:

I. Scatter read and check forward and reverse. This routine will read words from tape in either the forward or reverse direction and either store them in specified areas in core or check them against the contents of such areas. The film reading programs use only the checking facility, both forward and reverse.

II. Gather write forward. It is possible to write only in the forward direction, so this routine has only the one option. It will take words from specified areas in core storage and write them as an IBM record on tape. In addition, the routine will write an end-of-file record if entered at location wef.

III. Space. This routine will space the tape forward or in reverse a specified number of records.

IV. Rewind. This routine rewinds the tape.

As used, all of the routines use logical tape unit 1.

magtape routines 15 jan 64

```
/I.  scatter read and check forward and reverse

201/

src,        lio csa             /entry for check mag tape reverse
            sma 200
srr,        lio cda             /entry for read mag tape reverse
            dac pic             /save return address
            law rel             /set up to check for eob character
            dap ebx
            lac lm1             /set stp to decrement
            dac stp
brh,        law 372             /set last delay
            dap std
            lac (rir 6s         /set final rotates
            dac r5
            dac r6
            stf 5               /flag 5=1 indicates reverse
            jmp s8
sfc,        lio csa             /entry to check mag tape forward
            sma 200
sfr,        lio cda             /entry to read mag tape forward
            dac pic             /save return address
hl,         law mis             /don't check for eob character
            dap ebx
            lac (law 1          /set stp to increment
            dac stp
frb,        law 127             /set final delay
            dap std
            lac cno             /no-op the final rotate
            dac r5
            dac r6
            lac (ril 6s         /rotate order for forward
            clf 5               /flag 5=0 indicates forward
s8,         dac r1              /set the rotates
            dac r2
            dac r3
            dac r4
            dio d1              /set the store or compare order
            dio d2
            lac stl             /jmp in if transport running
            sza
            jmp fir
            law 200             /start transport forward or reverse
            szf 5
            law 240
            jda set
            ril 3s              /check for load point
            spi
            jmp frs
            law i 4000          /bypass 3 inches if at load point
```

```
        jda del
frs,    law i 505        /forward read starting time
        szf 5
brs,    law i 226        /backward read starting time
        jda del
fir,    idx fch          /first character indicator
        dzm ckr          /initialize
        dzm psc
        dzm noc
        xct stp          /set end test to fail
        dac mte
        dzm mpt
        lio i pic        /get first address or bypass number
        law nor          /set entry
        spi
        law byp
        dap lox
        spi              /bypass is first
        jmp sel
        dio mpt          /set pointer to first address
        idx pic          /set end check
        lac i pic
        dac mte
        idx pic
sel,    law i 7777       /look for 80 milliseconds
        dac lot
        clf 2            /clear flags and buffer
        clf 4
        clf 5
        clf 3
        mcb
lko,    szf 2            /wait for flag 2
lox,    jmp .            /go to normal entry or bypass entry
        isp lot          /count
        jmp lko
        jsp upc          /index pickup and stop transport
nch,    jmp .            /error halt for no characters on tape
rel,    law mis          /undo eob check
        dap ebx
        jmp sel
nor,    cli              /clear io and read in first 6 bits
        xct h5
        szf i 2          /10 norm-5 to speed up
        nop
        szf i 2          /15 norm-5 to speed up
        xct cno
        jmp ent          /away we go
top,    szf i 2
cno,    nop              /10 norm-5 to speed up
stp,    0                /law 1 or law i 1
        cli              /clear io and check for miss
        szf i 2
        jmp mis
        mrc              /read first six bits
        add mpt          /increment pointer
```

```
            sad mte             /check for end of memory block
            jmp swi
            dac mpt             /store the incremented pointer
ent,        szf i 2             /15 norm-5 to speed up
            xct cno
            szf i 2             /5 norm-20 to slow down
            xct h2
r3,         0                   /rotate io and check for miss
            szf i 2
ebx,        jmp .               /miss or relook
            mrc                 /read second six bits
nfc,        dzm fch             /indicate not first character
            law mis             /undo eob exit since 2nd character seen
            dap ebx
            xct h3
h2,         xct cno             /15 norm-5 to speed up
            szf i 2
            xct h2              /5 norm-20 to slow down
r4,         0                   /rotate io and check for miss
            szf i 2
            jmp mis
            mrc                 /read third six bits
r6,         0                   /final rotate or nop
            rcl 9s              /word into ac
            rcl 9s
d2,         0                   /store or compare
            jmp top
            jmp rdc
mis,        dio bdw             /record last character read and enter
            lac fch             /miss if not first character
            sza i
            jmp msn
            law 1700            /is first character 17?
            sas bdw
            jmp msn
            lac par
            sza
            clc
            szf 4
            cma
            sza i               /check for proper parity
            jmp msn
            xct std
            jda del
            jsp upc             /index pickup and stop transport since
                                /eof mark seen
efh,        jmp .               /halt because end of file found
swi,        lac i pic           /get next first address
            spa                 /if minus, it is number to bypass
            jmp bys
            szf i 2             /5 norm-20 to slow dow
            xct h2
```

```
rl,     0                /rotate io for next character
        szf i 2          /check for miss
        jmp mis
        mrc              /read second six bits
        dac mpt          /store new pointer
        idx pic
        lac i pic
        dac mte          /store new mte
        szf i 2          /5 norm-20 to slow down
        xct h2
r2,     0                /rotate io and check for miss
        szf i 2
        jmp mis
        mrc              /read third six bits
        idx pic          /step pickup for next block
r5,     0                /final rotate or no-op
        rcl 9s           /word into ac
        rcl 9s
dl,     0                /store or compare
        jmp stp          /loop
rdc,    dac bdw          /store the offending word
        idx ckr          /indicate a read check error
spc,    stf 5            /set flag 5 so 3 won't get set
msl,    szf i 5          /record a miss (flag 3) unless 5 on
        stf 3
        clf 2
msn,    mcb              /clear buffer for next character
        law i 5          /set for 200 μs look
        dac lot
msk,    szf 2            /start over when character seen
        jmp msl
        isp lot          /continue looking
        jmp msk
        xct stp          /update block count
        add blo
        dac blo
fsh,    lac i pic
        cma              /step pickup to return
        sza i
        jmp s4
        idx pic
        jmp fsh
s4,     idx pic
        szf 4
        jmp s6           /stop and error return if parity error
        szf 3
        jmp s6           /same if missed character
        idx pic          /set return for normal
        lac ckr
        sza
        idx pic          /return 1 past normal if check error
        lac sop
```

```
s7a,    dac stl         /record whether or not stopped and
        dzm sop         /reset to stop next time
        sza             /if not zero, don't stop
        jmp nod
std,    law i 0         /final delay
        jda del
        lio unt
        msm             /whoa
nod,    lio psc
        stf 2
        xct stp
        add mpt
        sas mte
        jmp s7
        spi i
        clf 2           /2 on unless psc + and ptr at end
s7,     law blo
        add unt
        dap dbl
        lac blo         /get current block count
dbl,    dac             /store block count for current unit
        mcs             /status bits in io for return
        jmp i pic       /return to calling program
s6,     cla             /force a stop
        jmp s7a
byp,    cli
        xct h5          /clear io and read first six bits
        lac i pic       /get the pass count in the ac
        szf i 2         /15 norm-5 to speed up
        xct cno
        szf i 2         /15 norm-5 to speed up
        xct cno
        szf i 2         /5 norm-20 to slow down
        xct h2
bys,    ril 6s          /check for miss
        szf i 2
        xct ebx         /miss or relook
h5,     mrc             /read second six bits
        dac psc         /times 3
        sal 1s
        add psc
        dac psc
        sza i           /bypass zero means end
        jmp spc
        idx pic         /step pic
        xct h3          /check for miss
        jmp mis
        mrc             /read third six bits
        idx psc         /count 2
        idx psc
```

```
psl,        isp psc             /count each character
            sma 200             /re-enter reading after countup
            jmp top
            szf i 2             /10 norm-5 to speed up
            nop
h3,         szf i 2             /5 norm-20 to slow down
            xct h2
            xct h3              /check for miss
            jmp mis
            mrc                 /read character
            xct nfc             /indicate first character passed
            jmp psl
set,        0
            dac exi
            law 3               /and off unit bits
            and unt
            dac unt
            law blo
            add unt
            dap gbl
gbl,        lac                 /get block count for current unit
            dac blo             /store it
s1,         lio unt             /select unit and interrogate
            msm
            mcs
            sma 200             /unit is ready if bit 0 = 0
            jmp rwi
            law 20
            and set
            sza i
            jmp s2
            ril 2s              /unavailable, is file protect on write
            spi i
            jmp uah
s2,         law i 764           /transport stopping time
            jda del
            lac par             /do the order after clearing
            sza                 /the status bits in io for return
            law 10
            ior set
            ior unt
            rcl 9s
            rcl 9s
            mcb
            msm
            mcs
s5,         jmp i exi
exi,        0
rwi,        ril s               /rewinding if bit 1 = 0
            spi i
            jmp s1
uah,        jsp upc             /undex pic
unh,        jmp .
unt,        1
```

```
cda,        dac i mpt
csa,        sad i mpt
bdw,        0
sop,        0
stl,        0
blo,        0
bk1,        0
bk2,        0
bk3,        0
par,        0
mpt,        0
mte,        0
pic,        0
del,        0                   /20 µs per count
            dap dex
a1,         nop
            isp del
            jmp a1
dex,        jmp .
upc,        dap ure
lm1,        law i 1
            add pic
            dac pic
            dzm sop             /record the stop
            dzm stl
            lio unt             /stop the unit and get status bits
            msm
            mcs
ure,        jmp .

/II. gather write forward

51/

gwf,        dac pic             /entry to write a block
            lac stl             /start writing immediately if
            sza                 /transport running
            jmp c14
            law 220             /start the transport
            jda set
            law las             /set to record max for current unit
            add unt
            dap dmx
            ril 3s              /check for load point
            spi
            jmp c1
            law i 6000          /extra delay for first block
            jda del
c1,         law i 301           /normal delay before writing
            jda del
```

```
c14,     lac i pic         /exit immediately if first word negative
         spa
         jmp c12
         dac mpt           /store first pointer and enter loop
         jmp c7
c4,      mwc               /write the second six bits
c11,     ril 6s            /rotate third six bits into position
         idx mpt           /increment pointer and check for
         sas mte           /end of block
         jmp c5
         lac i pic         /fetch new pointer and store
         dac mpt
         spa               /negative signifies end of list
         jmp c10
         xct c4            /write third six bits after delaying 5
c7,      idx pic           /fetch new end check
         lac i pic
         dac mte
         idx pic
c6,      lio i mpt         /fetch next word to write
         mwc               /write the first six bits
         xct c11           /rotate 2nd six into position, delay 5
         law i 1
         jda del           /delay 45
         jmp c4
c5,      xct . 1           /delay 25
         xct cno
         xct c4            /write third six after 5 delay
         xct c5            /delay 40
         xct c5
         jmp c6
c10,     mwc               /write the final six bits
         idx blo           /step block counter
dmx,     dac               /record max for current unit
c12,     idx pic
         law i 13          /delay 235
         jda del
         mcb               /clear writers
         mcs               /status bits to io
         stf 2             /error halt if end point passed
         ril 6s
         spi
         clf 2
         spi i             /force a stop if error
         dzm sop
         lac sop           /record a continue
         dac stl
         dzm sop
         sza               /don't stop if sop not zero
         jmp c13
```

```
fwh,        law i 474               /halting time
            jda del
            lio unt                 /stop the transport
            msm
c13,        mcs                     /status bits to io for return
            szf i 2                 /halt if end point passed
            jmp s7
eph,        jmp .                   /end point halt
wef,        dac pic                 /entry to write end of file
            lac unt                 /stop the transport
            jda set
            law 230                 /start write even parity
            jda set
            law i 4000              /3 inch gap
            jda del
            lio (170000             /write the end-of-file character
            mwc
            dzm sop                 /force a stop

las,        jmp c12 1               /enter last of write routine
mx1,        0
mx2,        0
mx3,        0


/III. space

14/

spt,        dac spx                 /save return
            lac i spx               /return immediately if count zero
            spa
            cma
            sza i
            jmp spr
            cma                     /set the block counter
            dac sco
            lio i spx               /sign of io indicates forward
            law sfr                 /or reverse - set for forward or
            spi                     /reverse
            law srr
            dap spo
            lac sco                 /ready to enter loop
spl,        add (1                  /see if only one block left
            sza                     /continue unless only one left
            idx sop
igt,        law i 620               /time past gap trash
            jda del
spo,        jsp                     /either sfr or srr
            777777                  /don't store anywhere
            nop                     /ignore errors
            isp sco
            jmp spl
```

```
spr,        idx spx
            lac blo              /block count in ac for return
            jmp i spx
spx,        0


/IV.  rewind

5/

rew,        dac pic
            law 340              /set to rewind
            jda set
            lio unt              /clear out the rewind order
            msm
            dzm blo              /set block count to zero
            jmp s7


start
```

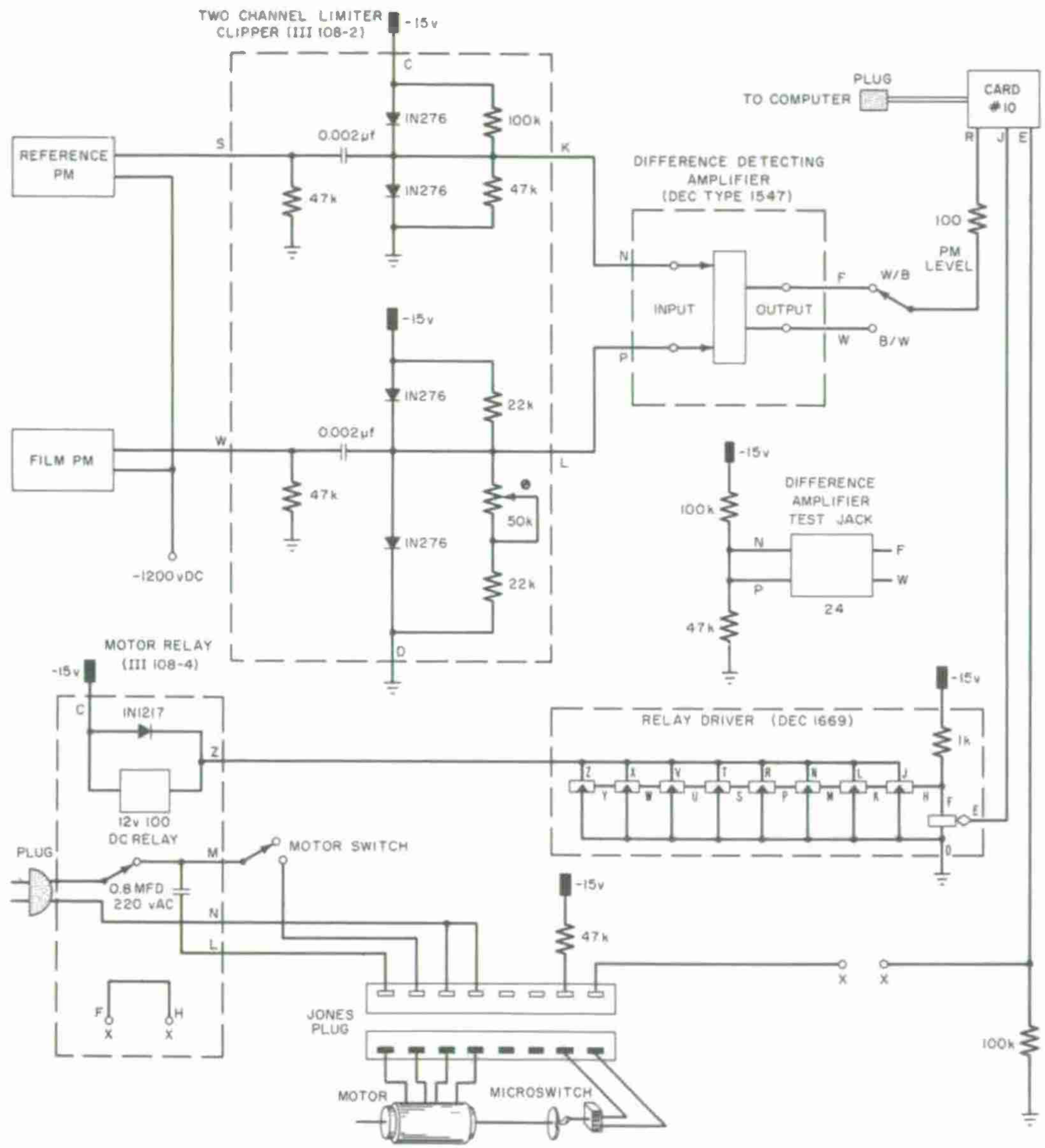
APPENDIX C

EQUIPMENT CIRCUIT DIAGRAMS

Figure C1 is a circuit diagram of the original film reader electronics and electrical connections. Since this drawing was prepared, the difference amplifier DEC Type 1547 has been replaced by a Type 1572, since the former is no longer being manufactured and the latter is said to be an improved replacement type.

Figure C2 shows the connections in the computer itself.

3-PA-2166



Film Reader To Computer Cable

Film Reader Card # 10 Pin	Wirecolor	Function	Computer Plug Pin	Destination in Computer
E	BL	Microswitch	39	IOM 04 3H21E1
F	BK	Gnd	40	
J	Y	Flag 6	41	F6 output 3H13V2
K	BK	Gnd	42	
M	G		43	
N	BK	Gnd	44	
R	BN	PM Level	45	3J24E
S	BK	Gnd	46	
U	BL	Gnd	47	
V	BK	(Spare) IOT 45	48	3J25E
X	R	(Spare) IOT 11	49	3J25J
Y	BK	Gnd	50	

Fig. C1 Film reader circuit diagram

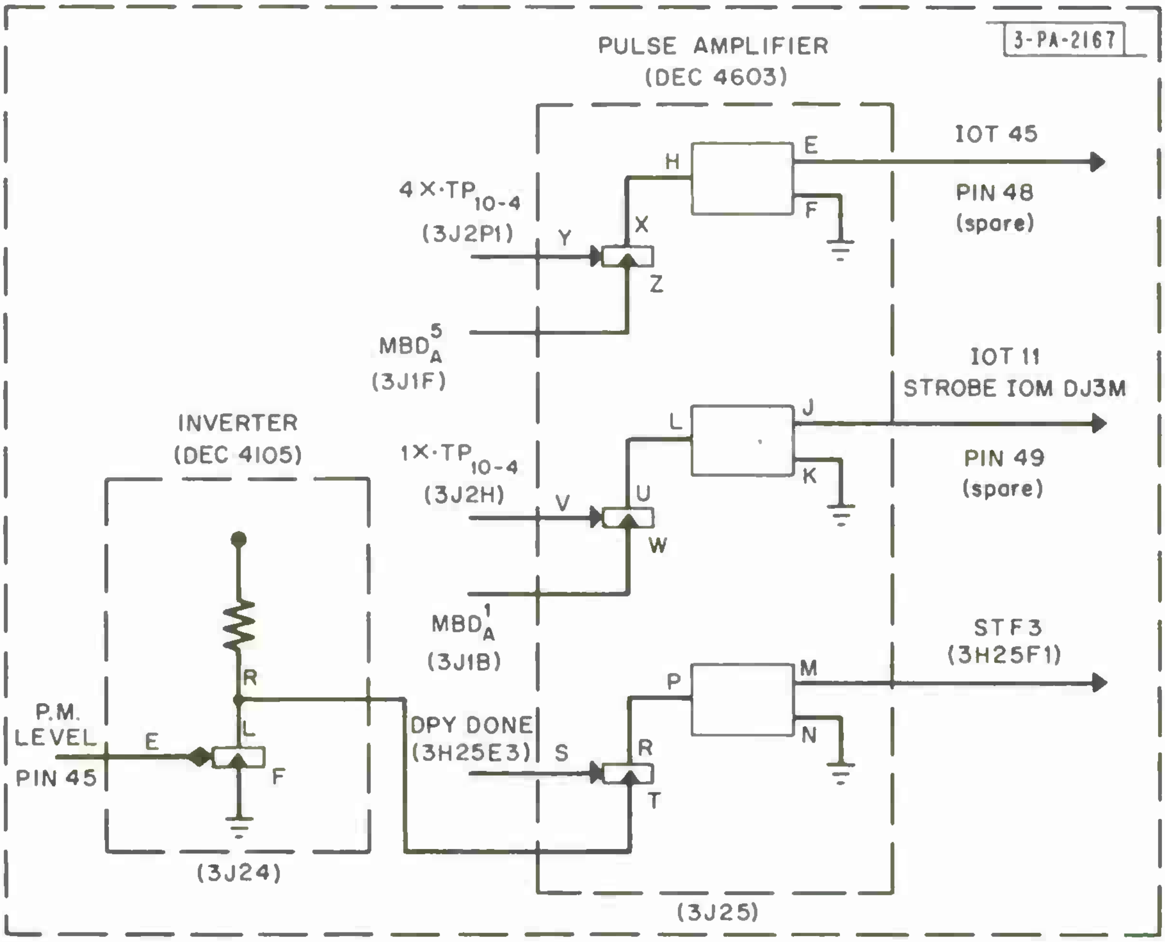


Fig. C2 Computer connections

APPENDIX D

MAGNETIC TAPE ROUTINES FOR AUTOMATIC EQUIPMENT

The set of magnetic tape routines given in the listing which follows was written by Bert Schafer to replace the earlier set when the Type 510 Tape Control and IBM drives were added to the installation. They are used in much the same manner as the Fletcher routines described in Appendix B except that the 'scatter-gather' feature has been dropped. Since the calling sequences were made to be of the same form as those for the earlier routines, the same block output routine can be used with either set.

There are six entries to the routines by the calling sequences listed:

1. Write

   jsp 70
   extended address of first register
   1 + extended address of last register
   unsuccessful write return
   end point return
   normal return

2. Read

   jsp 440
   extended address of first register
   1 + extended address of last register
   unsuccessful read return
   end of file return
   normal return

3. Space

   jsp 561
   number of records to space (+ = forward, - = backward)
   number of records too large return
   end-of-file return
   normal return

4. Write End-Of-File

   jsp 415
   end point return
   normal return

5. <u>Read Compare</u>

jsp 503
extended address of first register in reference block
1 + extended address of last register in reference block
unsuccessful read return
end-of-file return
no compare return

6. <u>Rewind</u>

jsp 633
return

Note that it is not possible to read or write in the reverse direction with this equipment.

schafer tape package for ibm equipment   17 jul 64

```
/symbols omitted from this version

swi=722046
swo=726046

5/
            0               /density, 0 for 200 cpi, 1 for 556, 2 for 800
            1               /unit, 0 thru 7
            0               /parity, 0 odd, 1 even
            0               /rewind indicator, 0 for no, 4000 for rewind
            0               /+ means start of function, - otherwise
            0               /record pointer
            0               /location of jsp to tape package
            0               /select instruction

            repeat 53,0     /table of functions

            jda 373         /initialize rwi, js, fup, tcu-unit free sr

            law i 3
            dac 720         /init blank tape counter
            law i 3
            dac 721         /init rewrite counter

            jsp 201         /go to tcu-unit free sr, setting sof to +0
            731071

            jsp 267         /process calling sequence for data channel

            sia
            lio 712
            swo             /swc out

            jsp 333         /execute function, test for acceptance
                            /and completion
            1               /increment record count
            jmp 116         /error return
            law 5           /no error--go to normal return
            add 13
            dap 115
            lac 373
            ral 1s
            sma
            lem
            jmp .           /return to main program

            jsp 166         /go to see if any errors beside end point

            jmp 123         /non-end pt errrors

            cli             /indicates end pt not during wbt
```

```
        law 4
        jmp 107          /go to end pt return in main program

        jsp 201
        730471           /basic backspace function

        jsp 332          /disconnect data control and execute function
        -1               /decrement record count

        jmp 147          /error return from backspace after write
        isp 721          /normal return
        jmp 75           /rewrite

        law i 10
        dac 704          /set counter to write blank tape
                         /8 times for a total of 6 inches

        jsp 201
        731071           /basic write mtf
        jsp 332          /disconnect data control and execute function
        -0               /do not change record count
        jmp 153          /error return from write blank tape
        isp 704          /normal return
        jmp 134

/6 inches blank tape written

        isp 720
        jmp 73           /init rewrite counter

        law 3
        jmp 107          /go to unsuccessful return in main program

        jsp 407          /save state reg and initial loc counter
        lio 672
        jsp 356
        jmp 130          /go to normal return

        jsp 166          /blank tape error
        jmp 157          /there are non-end pt. errors
        lio 706          /make IO negative implying end pt during wbt
        jmp 121          /go to end pt return in main program

        jsp 407          /save state reg and init loc counter
        lio 673
        jsp 356

        ral 7s
        spa
        jmp 155          /end pt
        jmp 141          /continue write operations

        dap 174          /routine to determine if other errors
                         /beside end pt
        swap
        and 714          /mask out tioe bit and all non-error bits
        sza i
        idx 174          /no other errors beside end pt
        jmp .
```

/tcu-unit free sr

```
        lio 706
        dio 11          /setting sof neg means not start of function
        dap 245
        jmp 224

        dzm 11          /setting sof pos means start of function
        dap 245         /set return to 1+ loc(jsp tcz)
        dap 210
        nop
        nop
        lac 7
        sal 7s          /move parity indication into bit 10
        ior .           /construct function with correct parity
        nop
        dac 336         /store into execution routine
        idx 245         /set return to 2+ loc(jsp tcz)

        law 1
        add 5
        sal 3s
        ior 6
        sal 6s
        ior 10
        ior 706
        dac 14          /store constructed select instruction

        lac 662
        dac 705         /set counter to -66666 decimal
        sfc
        jmp 231
        jmp 243
        jsp 407
        isp 705
        jmp 226
        cgo             /tcu not free within 5 secs

        lio 663
        lac 11
        spa
        lio 664
        jsp 356         /go to error typeout specifying cb1 or cb2
        jmp 367         /go to halt again

        lac 11          /tcu is free--is this start of function?
        spa
        jmp .           /normal exit
        xct 14          /select tape
        rsr
        rir 1s
        spi
        jmp 246         /tape is rewinding

        lac 662
        dac 705         /set counter to -66666 decimal
        rsr
        spi 1
        jmp 245         /normal exit
```

```
        jsp 407         /save state reg and init loc counter
        isp 705
        jmp 255
        cgo             /unit not free within 5 secs
        lio 665
        jsp 356
        jmp 246         /try again


/sr to load data channel

        dap 322         /set up return
        lac 373
        dap 272
        lac .           /pick up reg containing 1st address
                        /of output or input block
        dac 711
        spa
        jmp 327         /illegal calling sequence
        sub 667         /(1st address) - (last address of package)
        spq
        jmp 323         /illegal except for write
        add 667         /restore 1st address
        sub 670         /(1st address) - 040000
        sma
        jmp 327         /illegal calling sequence
        idx 272
        xct 272
        spa
        jmp 327         /illegal calling sequence
        sub 711
        dac 712         /store tentative word count
        spq
        jmp 327         /illegal calling sequence
        add 711         /restore 2nd address
        sub 670         /(2nd address) - 040000
        szm             /2nd address may be ≤ 040000
        jmp 327         /illegal calling sequence
        lio 711
        jmp .

        lio 336         /pick up function
        ril 9s
        spi 1
        jmp 301         /legal for write

        lio 671
        jsp 356         /type out error message for illegal
                        /calling sequence
        jmp 145         /go to unsuccessful return
```

/routine to execute function and wait for completion

```
        sdf
        dap 355         /set up return
        dap 345
        xct 14          /select tape to clear indicators
                        /in state register
        0               /function to be executed is prestored here

        sfc
        jmp 344         /function accepted
        lio 666
        jsp 356         /function not accepted--type error message
        jmp 335         /try again

        jsp 175         /function accepted -
                        /is tcu free in less than 5 secs?
        lac .           /pick up record number increment
                        /or decrement
        add 12
        dac 12
        idx 355

        rsr
        ril 1s
        spi i
        idx 355
        jmp .
```

/error typeout routine

```
        dap 372         /set up return
        ril 6s
        tyo
        ril 6s
        tyo
        ril 6s
        tyo
        lio 713         /put code for carriage return
                        /in right 6 bits
        tyo

        lac 707
        lio 710
        hlt
        jmp .           /return to program which called
```

```
/initialization routine

                            /1+loc(jsp to tape package)
            dap 406         /set up return
            dzm 10

            lac 373
            dap 563
            nop
            sub 456         /1
            dap 13
            nop
            nop
            nop
            jmp .


            dap 414
            rlc
            dio 710
            rsr
            dio 707
            jmp .


/write end of file section

            jda 373
            jsp 201
            733271          /write eof function
            jsp 332         /disconnect data channel and
                            /execute function
            1               /increment record count
            jmp 425         /error return
            law 2
            jmp 107         /return to normal return of main program

            jsp 166
            jmp 431         /there are non-end pt errors
            law 1           /only end pt
            jmp 107         /go to end pt return of main program

            jsp 407         /save state
            lio 676
            jsp 356

            ral 7s
            spa
            jmp 427         /go to end pt return of main program
            jmp 423         /go to normal return of main program


/section for read and read-compare

            dzm 703         /set read compare indicator to straight read
            jda 373

            law i 3
            dac 717
```

```
jsp 201
731471

jsp 267          /process calling sequence
lac 703
sza
jmp 506          /read compare

sia
lio 712
swi              /swc in

jsp 333          /execute function, test for
                 /acceptance and completion
1                /increment record count
jmp 467          /error return
rir 5s           /no error--move eof bit into sign
spi
jmp 121          /end of file return
lio 703
spi i
jmp 106          /normal return
jmp 514          /read compare


jsp 201
730471           /basic backspace function

jsp 332          /disconnect data control and
                 /execute function
-1               /decrement record count
jmp 477          /error return from backspace after read

isp 717          /normal return from backspace
jmp 444          /re-read

jmp 145          /go to unsuccessful return in main program

jsp 407          /save state
lio 674
jsp 356
jmp 474          /normal return

lio 472          /-1
dio 703          /set rci to specify read compare
jmp 441


idx 712          /index word count
sub 700          /(1+specified word count) - (1024 decimal)
szm
jmp 327          /illegal calling sequence
lio 702          /start of read compare buffer
jmp 452
```

```
/successful read-----go to do comparison

            law i 1
            add 712
            dac 712          /restore wc to length of core block
            cma
            dac 704          /set counter to -(core block length)
            rlc
            swap
            and 713
            sub 702          /number of words read is now computed
            sas 712
            jmp 547          /blocks not equal
            cli              /set IO to +0


            lac 711
            dac 715
            lac 702
            dac 716
            eem
            lac i 715
            sas i 716
            jmp 106          /go to no compare return in main program
            idx 715
            idx 716
            isp 704
            jmp 536

/comparison OK

            law 6
            jmp 107          /go to comparison ok return in main program

            sub 712          /blocks not equal
            spa
            jmp 554          /tape block < core block
            lio 472          /tape block > core block
            jmp 531          /set IO to -1, ct1 already set properly


            lio 137          /words read < core block, set IO to -0
            add 712          /restore number of words read
            cma
            dac 704          /set counter to -(length of tape block)
            jmp 531
```

/section for space function

```
        jda 373
        lio 470         /basic backspace function
        lac .           /pick up plus or minus the number
                        /of records to be spaced
        sma
        lio 445         /basic forward function
        dio 601
        lio 472         /-1
        sma
        lio 456         /+1
        dio 613
        sma
        cma
        sad 137         /-0
        jmp 121         /zero spacing required, go to normal return
        dac 704         /counter contains  -(no. of records to space)

        jsp 201
        0               /basic function is preset
        lac 12          /pick up record pointer
        sza
        jmp 612         /record pointer not 0 - perform
                        /spacing function
        xct 563         /pick up ± number of records to space
        sma
        jmp 612         /perform forward spacing function
        lio 704
        jmp 423         /go to record 0 return in main program


        jsp 332         /disconnect data control and execute function
        -1              /-1 or +1
        jmp 624         /error return
        rsr             /no error--bring state into IO
        rir 4s          /rotate eof bit into sign
        spi
        jmp 630         /eof found
        isp 704
        jmp 600         /space some more
        jmp 121         /go to normal return

        jsp 407         /error from spacing - save state reg and
                        /init loc counter
        lio 675
        jsp 356         /error typeout
        jmp 615         /normal return

        idx 704
        lio 704
        jmp 145         /go to end of file return
```

/section for rewind

```
            jda 373
            law 4000
            dac 10          /set rewind indicator to rewind state

            jsp 201
            730471          /basic backspace function
            swap
            rar 6s
            nop
            spa
            jmp 651         /at load point

            jsp 332         /disconnect data control and
                            /execute function
            0
            jmp 653         /error return
            dzm 12          /no error--reset record pointer

            jmp 427         /go to normal return in main program

            jsp 407         /save state reg and init loc counter
            lio 677
            law 1 700       /mask to remove parity and
                            /density from state indicators
            and 707
            sas 701
            jsp 356

            jmp 651
```

/constants, temp storage, masks, etc

```
decimal
            -66666
octal

            text (cb1(
            text (cb2(
            text (ub (
            text (fna(
            721             /last reg of program
            040000
            text (ics(
            text (ebw(
            text (ebt(
            text (ebr(
            text (esp(
            text (ewf(
            text (erw(
```

```
        2000
        600001
        036000       /start of read compare buffer
        0            /read compare indicator, 0 for
                     /straight read, - for read compare

        0
        0

        720070
        0
        0
        0
        0


        177777
        370001


        0
        0


        0            /reread counter
        0            /blank tape counter
        0            /rewrite counter

start
```

APPENDIX E

CALIBRATION OF PROGRAMMABLE FILM READER

I. Six test patterns were chosen to evaluate the performance of the PFR, and are displayed in Figs. E-1a — E-6a.*

(E-1a) a straight line

(E-2a) an inverted "V" with angle of about 69°

(E-3a) an inverted "W" with all angles about 60°

(E-4a) two inverted "V's", each with angle of about 28°

(E-5a) a descending staircase; and

(E-6a) three rectangular pedestals of decreasing altitude and constant width.

Because a sharp angular trace should tax the system more than one that changes direction slowly, these traces, with their vertical and horizontal segments and triangular waves of varying steepness, seemed to be good diagnostic tools. In addition, note that patterns E-1a — E-4a are bilaterally symmetric, affording a check on whether horizontal distortion is introduced due to the curvature of the scope face. In the same fashion, Figs. E-5a and E-6a afford a check on potential vertical distortion.

All traces are equally long, being bounded left and right by the same pair of calibrated rectangular pedestals. The vertical lines at the side of the traces were intended merely as reference marks for the operator of the PFR. When the traces were prepared, a camera-alignment error resulted in a slight tilting of the traces on the film. The PFR, of course, reproduced this tilt, so that the right side of each trace appears somewhat higher than the left.

Each trace was photographed twenty times, but only one of each kind was later read by the film reader. Preliminary tests indicated that, for each series of 20 frames

*Figs. E-1b through E-6b, shown on facing pages, represent photographed redisplays of the trace readings and are described in a later paragraph.

there was no greater variability from frame to frame than there was in repeated readings of a single frame. As a consequence, the calibration tests described below used only one of the twenty frames available for each of the six patterns.

The length and width of trace segments were measured with a Bausch and Lomb toolmaker's microscope. This microscope is scaled to .0001 inch and reset accuracy was determined to be $\pm$ .0002 inch. These measurements were to be used for an error analysis of the PFR readings. However, as Figs. E-1—E-6 show, the error involved in the automatic film reading was so small as to render such an analysis unnecessary. Several of the measurements were used to obtain an accurate estimate of the number of scope points corresponding to an inch of film. This estimate, 670 scope points per inch, was used to corroborate the PFR count of 12 scope points for the width of traces used in this study.

Using a special modified version of the Series 1 program, described in Sec. II.A, the traces were read by the film reader. One of the modifications was to change the film advance subroutine to handle framed film. Every other scope coordinate was sampled in both the vertical and horizontal direction. All traces were read with an f-stop of slightly more than 5.6 for the signal lens and an f-stop of 11 for the reference lens. Each trace was read ten times in order to estimate the variability inherent in the system.

The high voltage supply to the photomultipliers, the scope intensity level, and the focus adjustment were set to yield a high fidelity reproduction of the trace. In other words, the system was tuned so that the redisplay was visually determined to be satisfactory. Preliminary tests had shown that, unless care were taken in focusing and in the setting of voltage and intensity levels, misreadings resulted, such as peak-flattening and trails at the edges of square waves.

The digitized amplitudes of points read were recorded on magnetic tape, retrieved, and examined. The disparity between two readings of the same trace was seldom greater than two ordinate units, and was usually zero or one. The system, in other words, consistently reported essentially the same ordinate value in multiple readings of a single trace.

The points read by the PFR were redisplayed on the oscilloscope by the tape viewing program and photographed. In order to facilitate comparison of the film traces with their associated oscilloscopic redisplays, prints were made both from the test film and from the directly photographed redisplays. Film traces and their associated redisplays are presented on facing pages in Figs. E-1 through E-6.

The radar scan program is designed to track the trace and to calculate its center by averaging the ordinates of its upper and lower edges. How well it does this was examined by the following procedure. The points read by the PFR were plotted on graph paper and mounted on a vertical surface normal to the beam of an ACME 35 mm projector. The appropriate film trace was then projected onto the plot of what had, in fact, been read. This was done for each of the six patterns. The plotted points and their superimposed projections are shown in Figs. E-7 through E-12. It is apparent that the PFR does an excellent job. Peaks are well-delineated, edges do not trail off systematically, and the center of the trace seems to be consistently read to an accuracy of one scope point.

II. This calibration test has shown that the automatic film reading system is basically sound. Some of the observed disparities between the film traces and the system's output, moreover, may not be the fault of the reader at all, but rather of variables such as imperfect positioning of the film in the film gate when reading is underway. In any case, with film of good quality disparities are small.

III. Each of the six test patterns was read ten times by the PFR. For each pattern, two of these readings were selected for comparison. The absolute value of the difference between corresponding amplitude readings was recorded and all differences were collected in the frequency table shown below.

		Absolute Difference				
		0	1	2	3	>3
Pattern I	Straight Line	211	45	–	–	–
Pattern II	Inverted V	179	63	10	4	–
Pattern III	Inverted W	161	86	9	–	–
Pattern IV	Two inverted V's	42	159	39	10	6*
Pattern V	Staircase	200	50	2	3	1*
Pattern VI	Pedestals	21	206	24	3	2*

For every paired comparison at least 93.8% of all points showed amplitude differences no greater than two units, and at least 97.7% of all points showed amplitude differences no greater than three units.

* Of the differences greater than three, all of those for Patterns V and VI and two of those for Pattern IV represent slight differences in alignment rather than inherent variability. All of these cases involve transitions to or from a horizontal trace portion: one reading gave the amplitude of a terminal point of this horizontal portion, while the other gave the amplitude of the first point beyond it.

-PA-1802

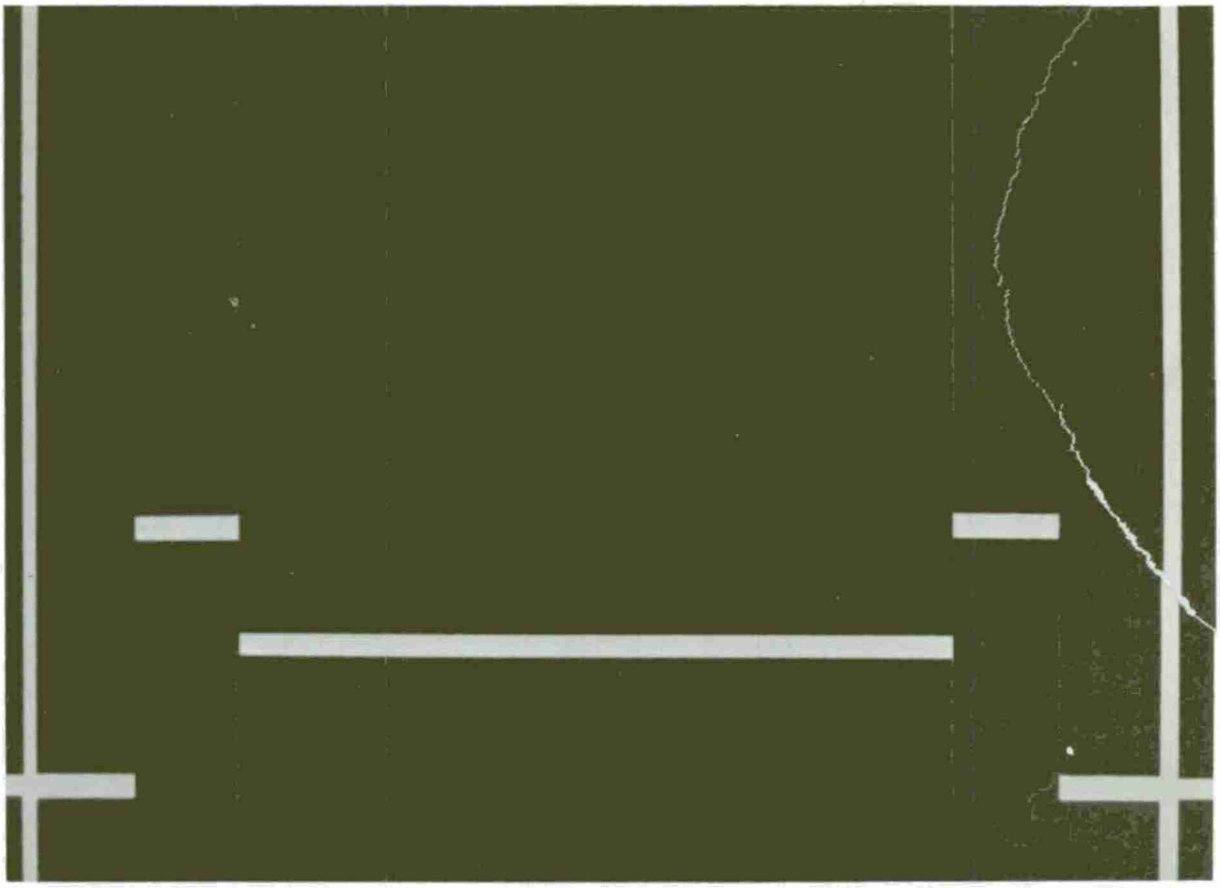


Fig. E-1a Print from film

P115-172

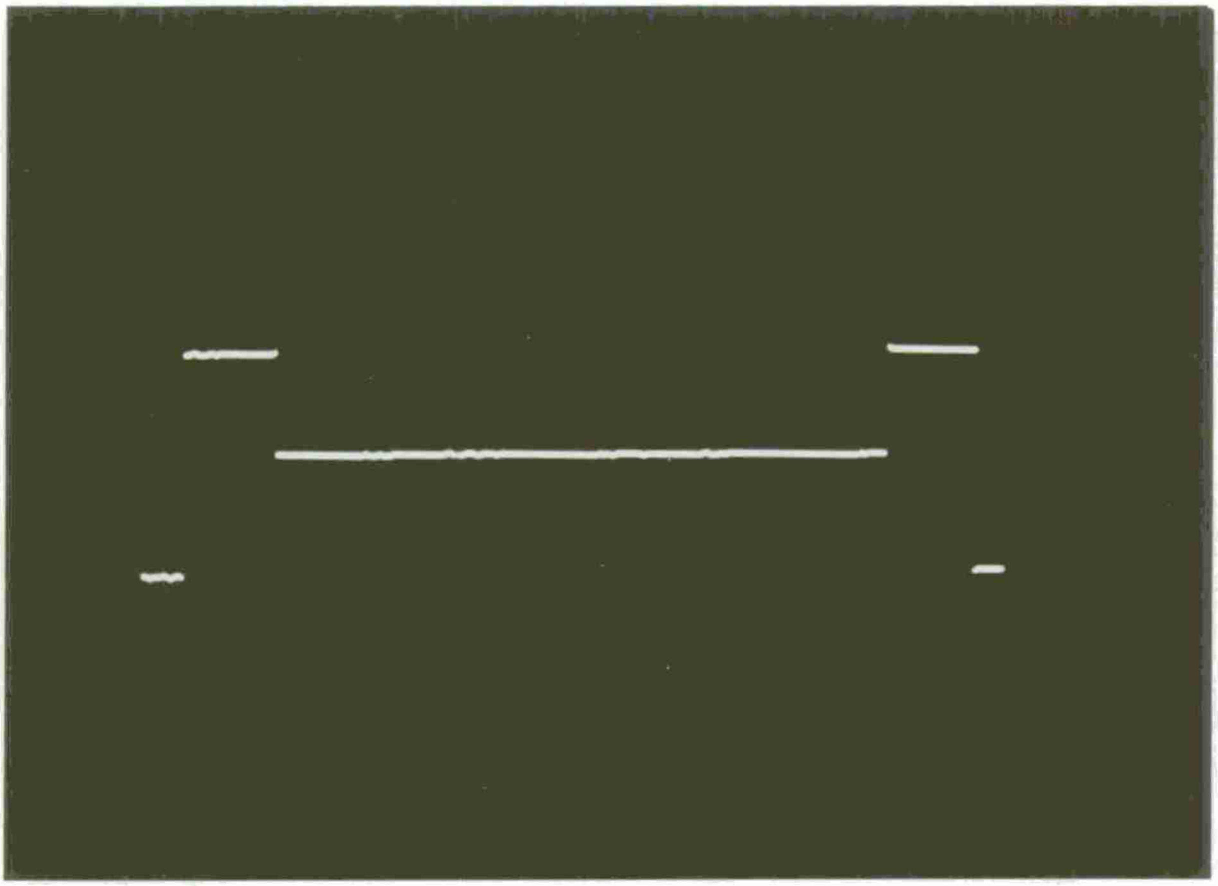


Fig. E-1b Print from redisplay

-PA-1803

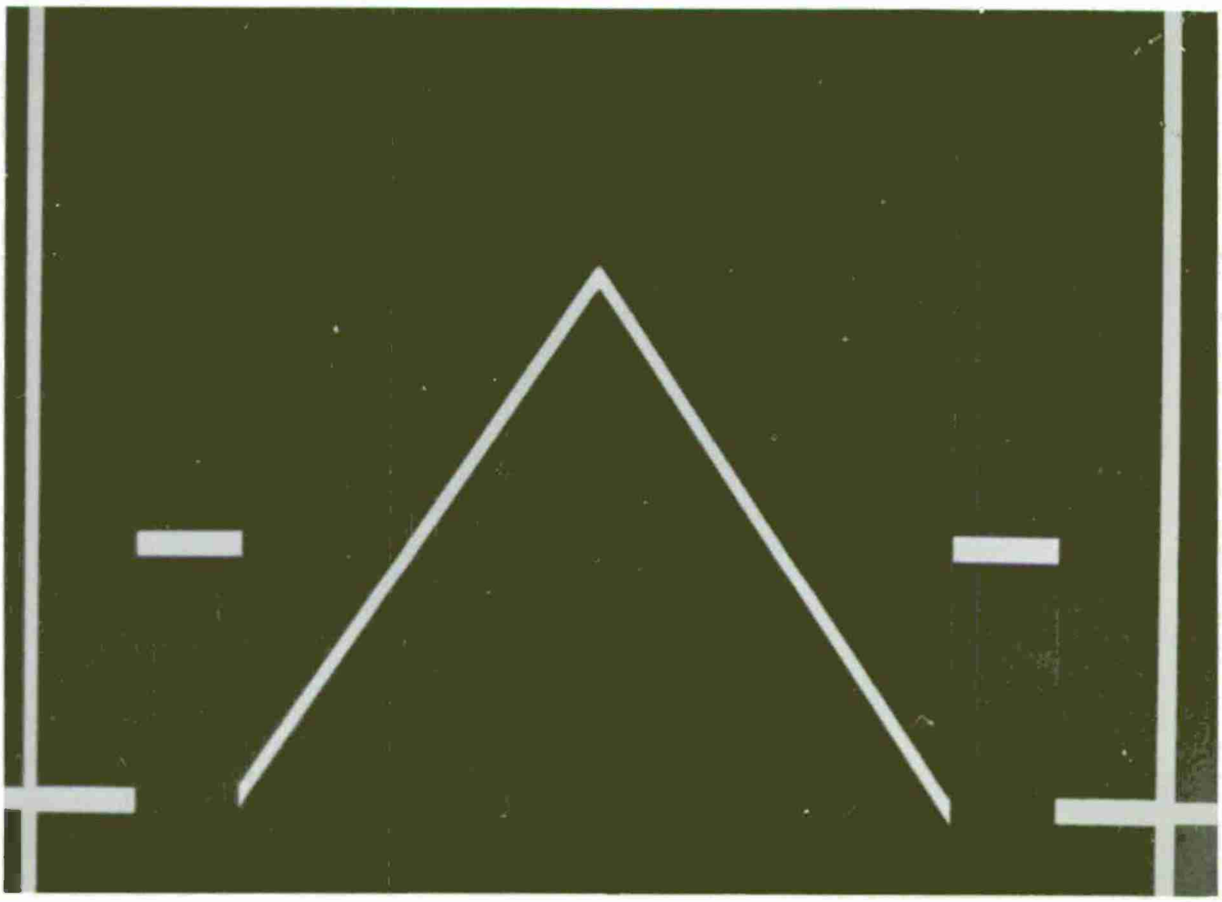


Fig. E-2a Print from film

P115-174

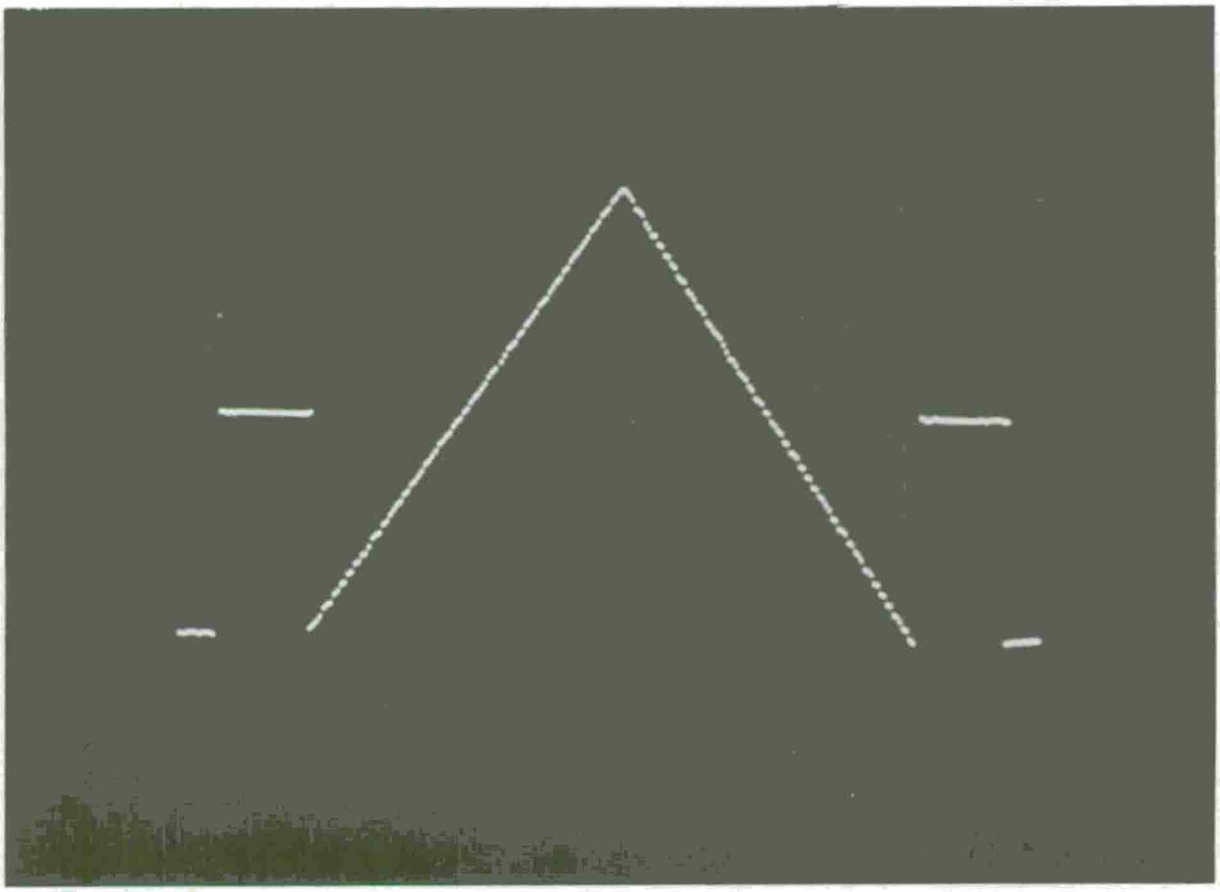


Fig. E-2b   Print from redisplay

-PA-1804

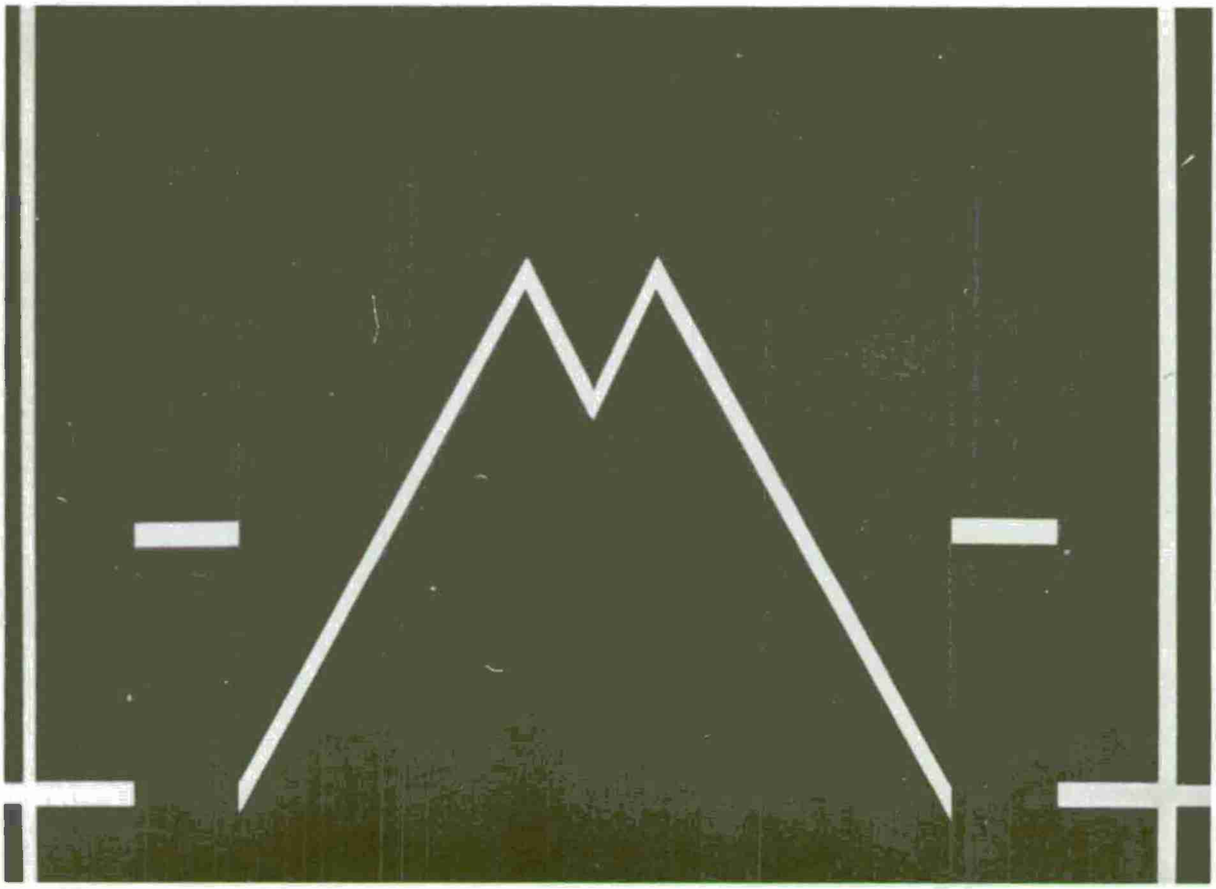


Fig. E-3a Print from film

P115-175

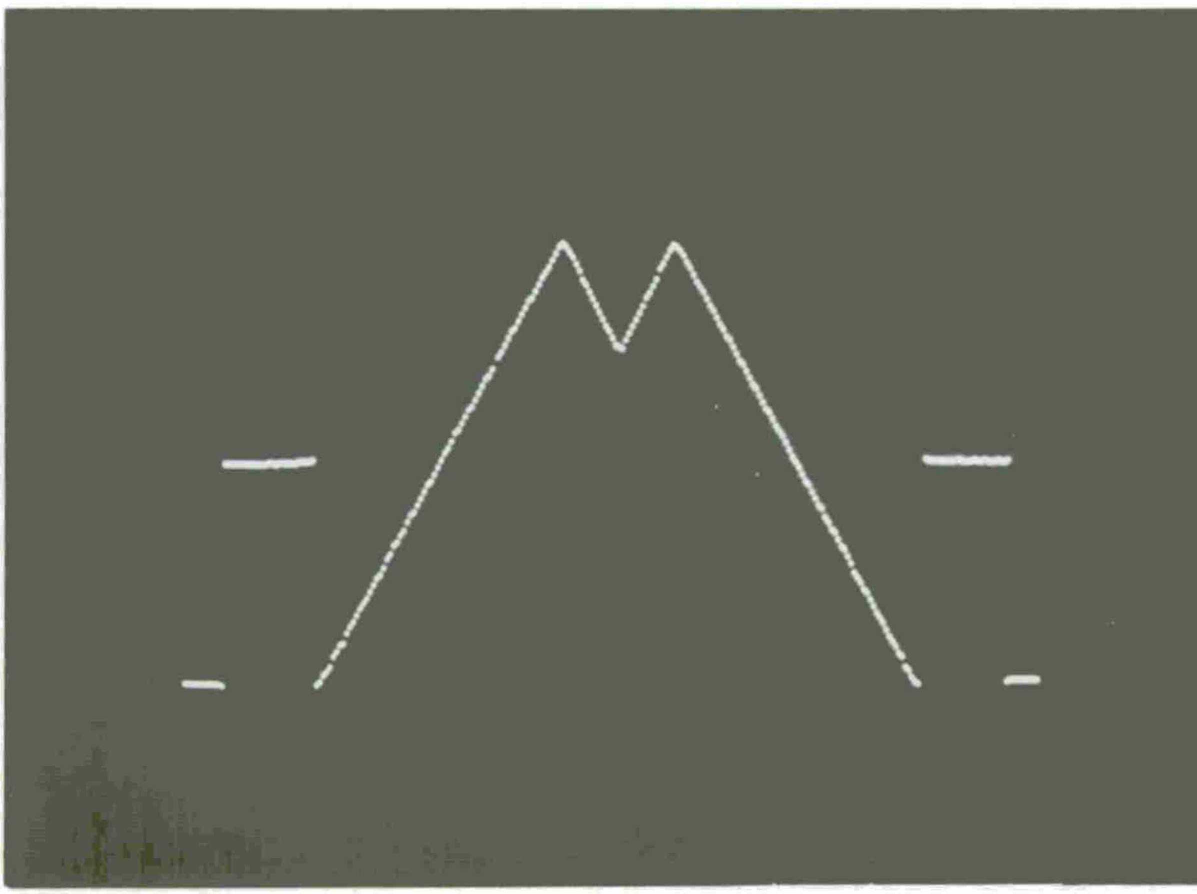


Fig. E-3b   Print from redisplay

-PA-1805

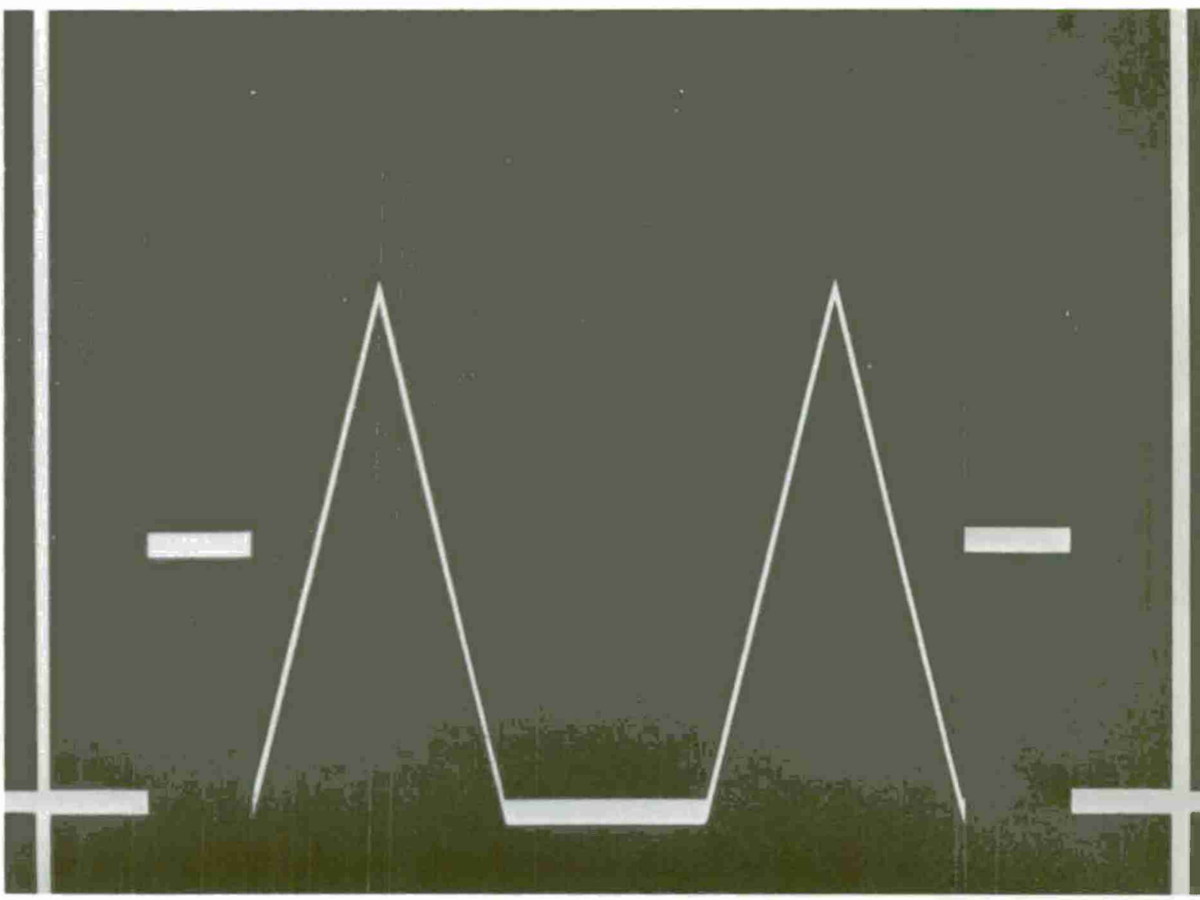


Fig. E-4a   Print from film

P115-176

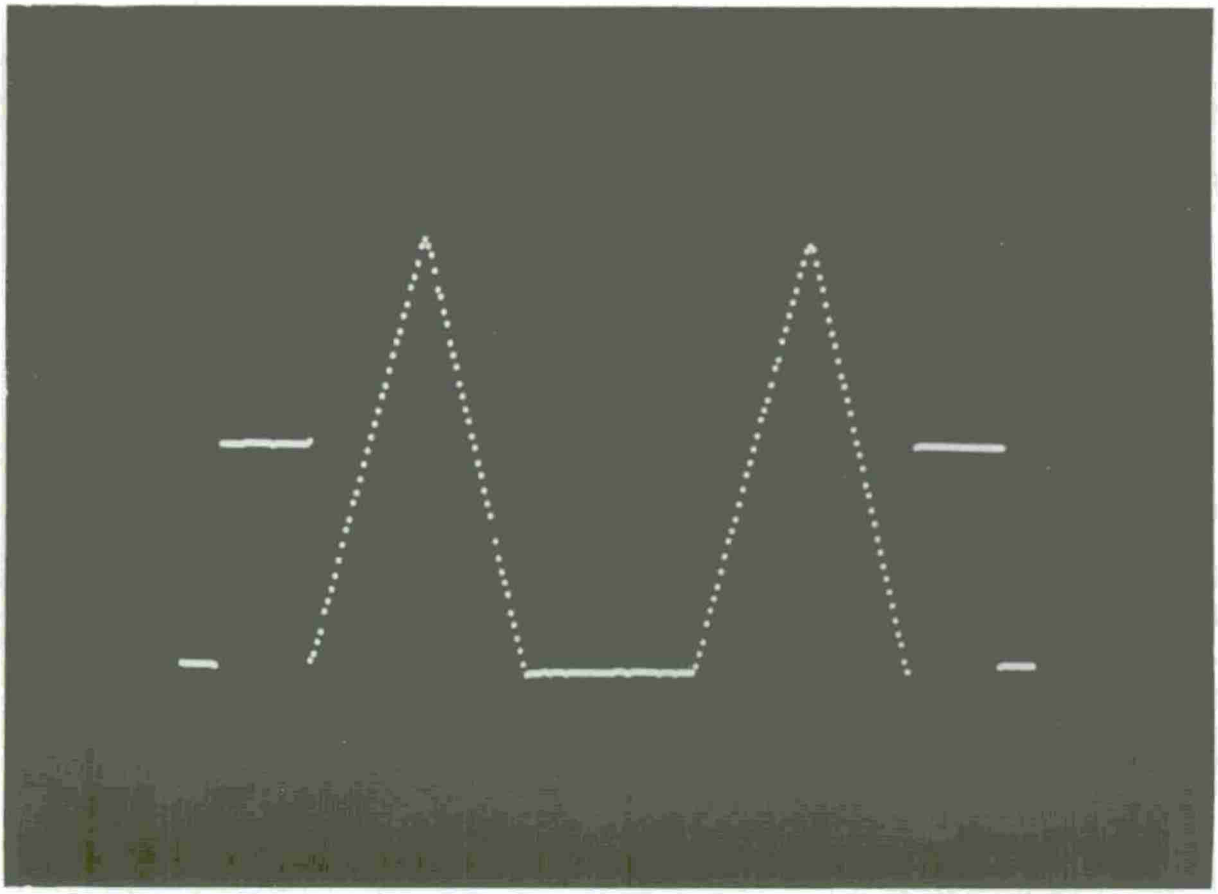


Fig. E-4b   Print from redisplay

-PA-1806

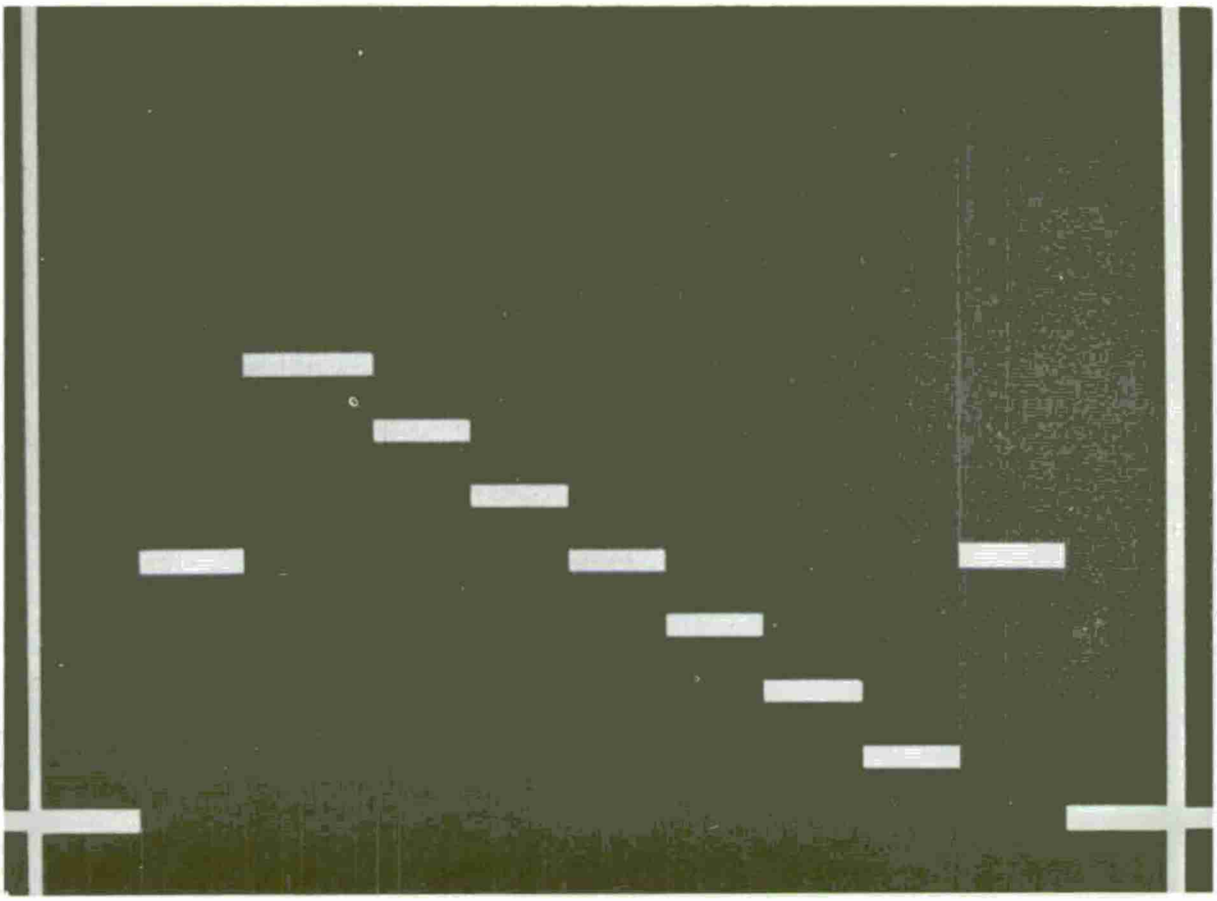


Fig. E-5a   Print from film

P115-173

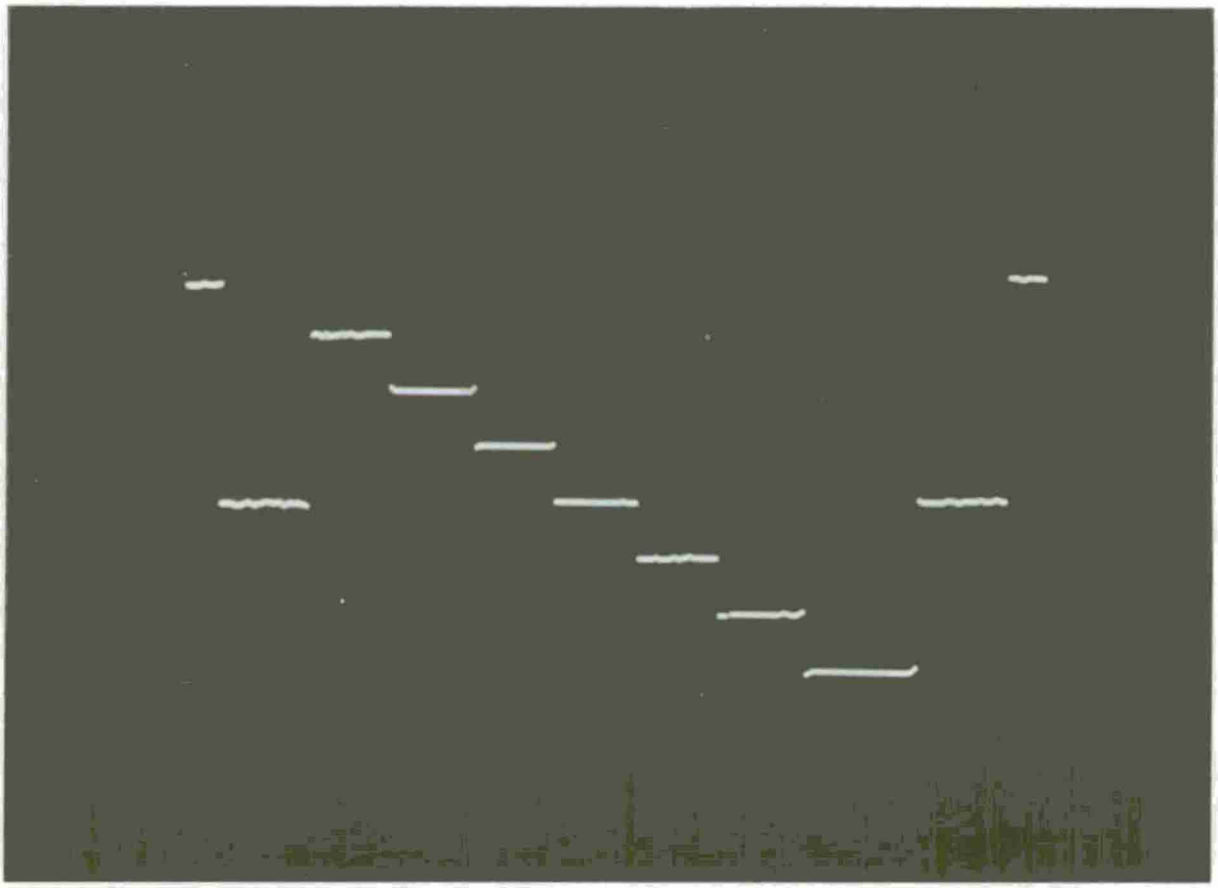


Fig. E-5b Print from redisplay

-PA-1807

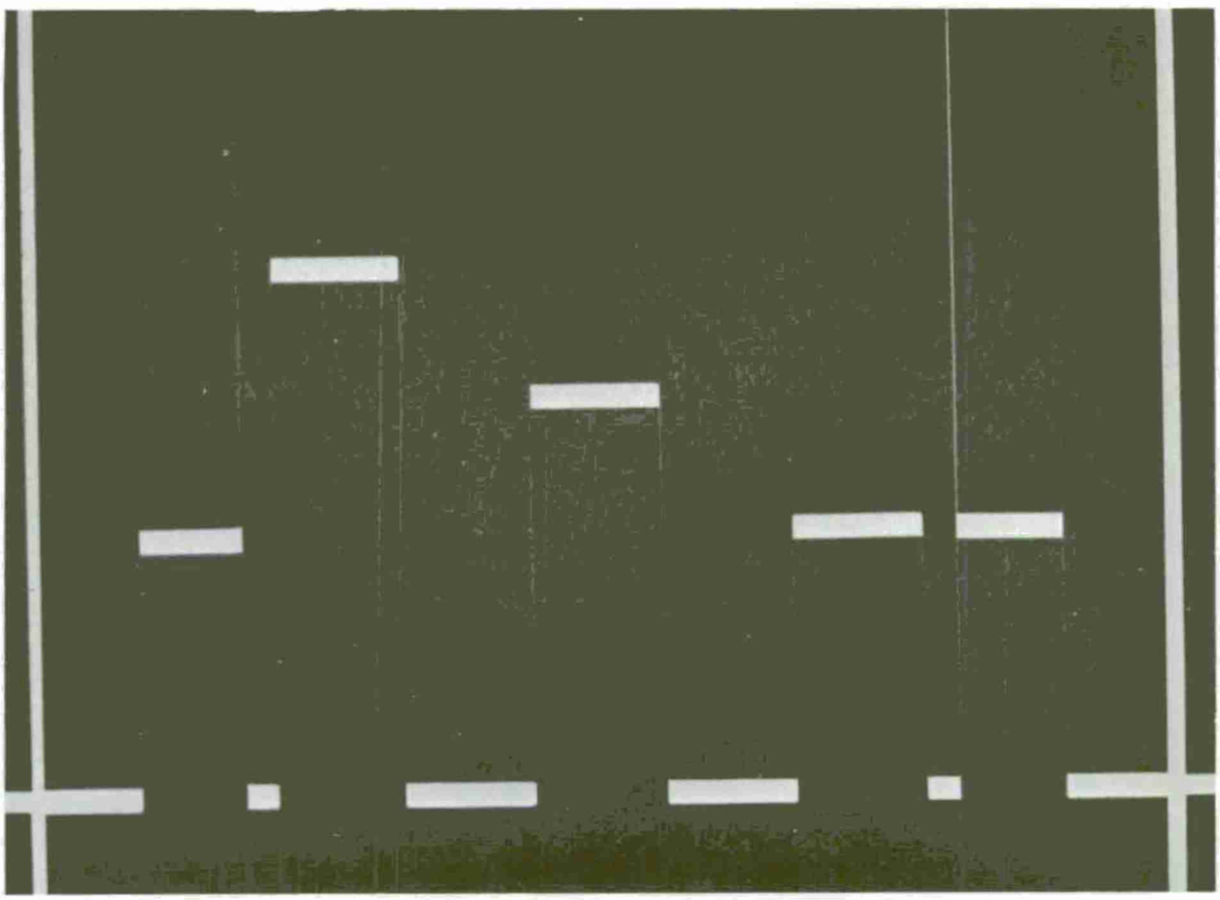


Fig. E-6a Print from film

P115-177

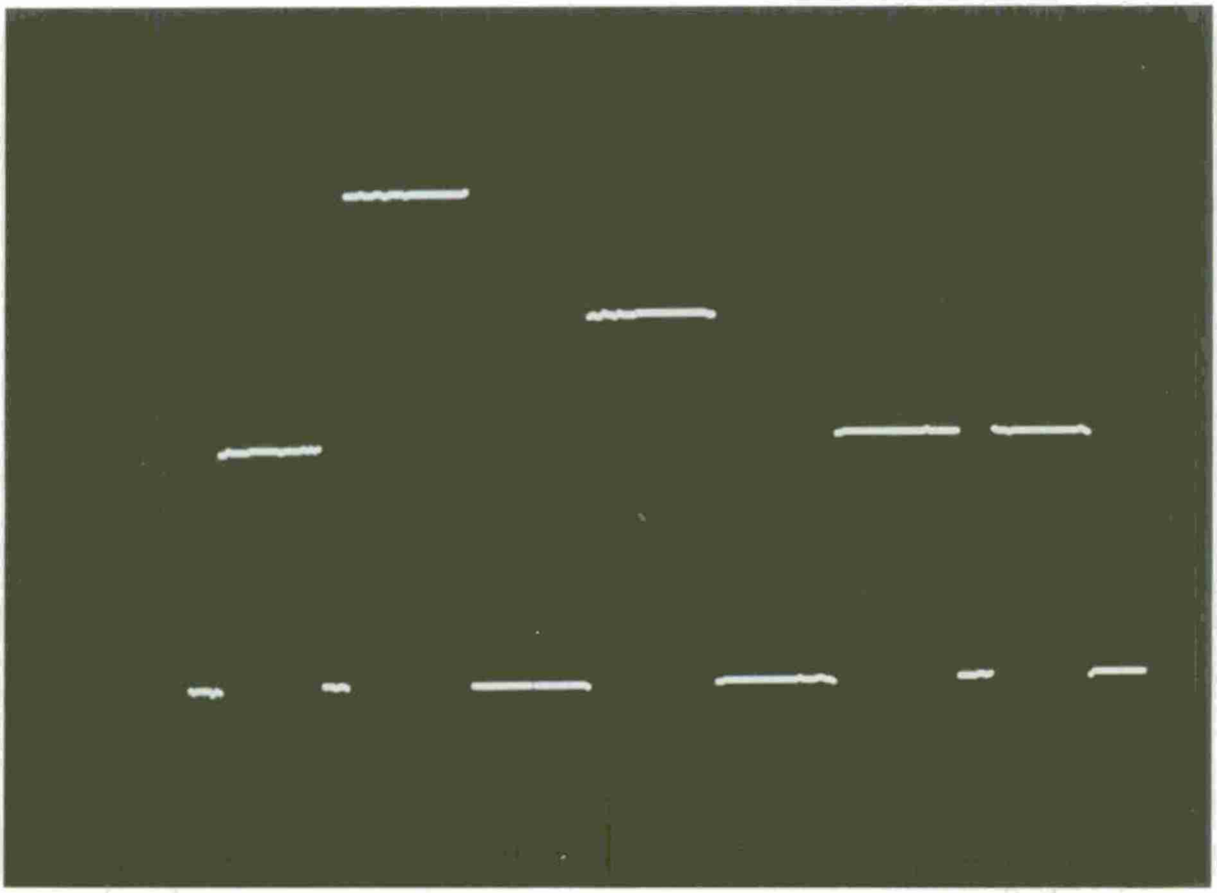


Fig. E-6b Print from redisplay

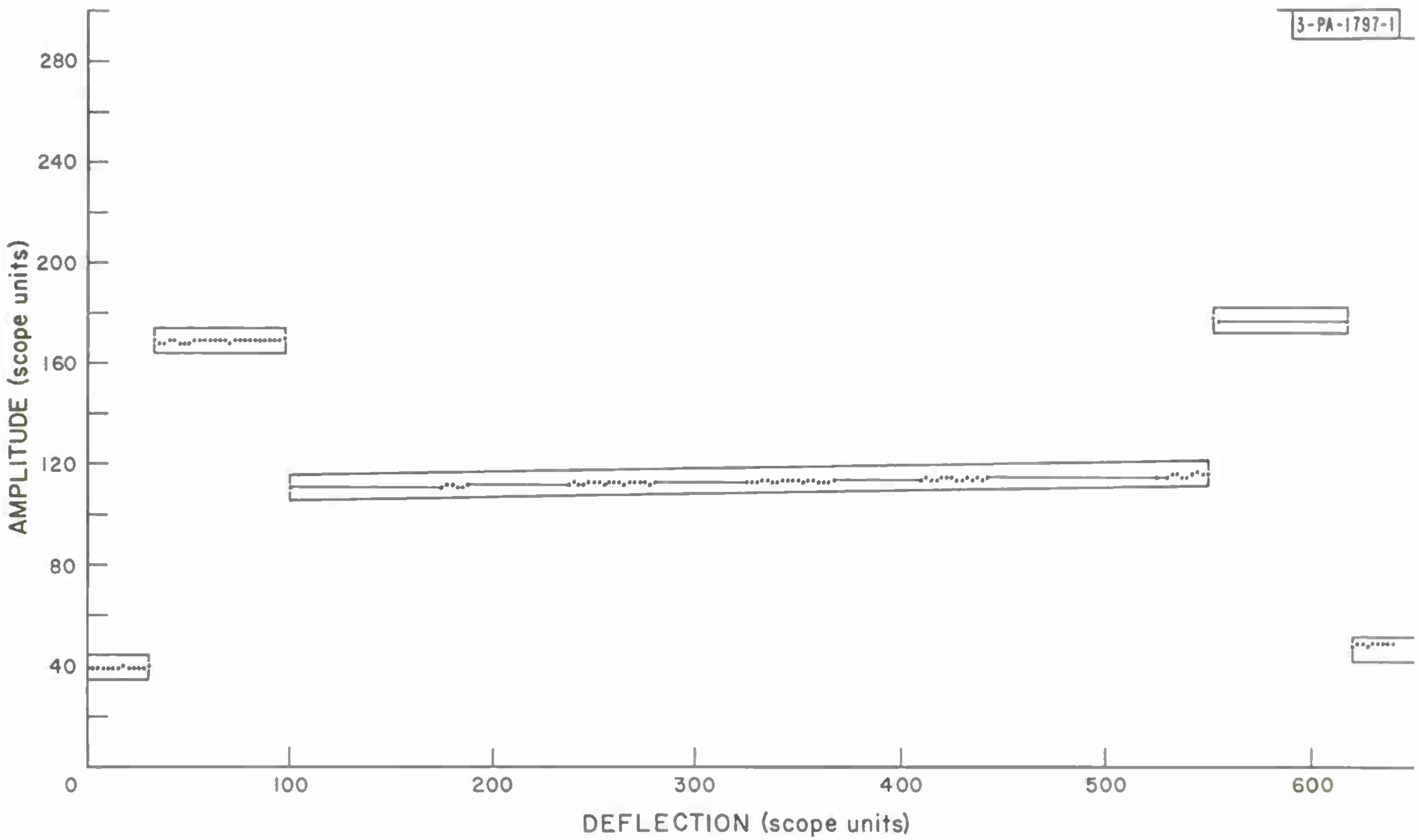


Fig. E-7 Graph of computer output superimposed on projection of film

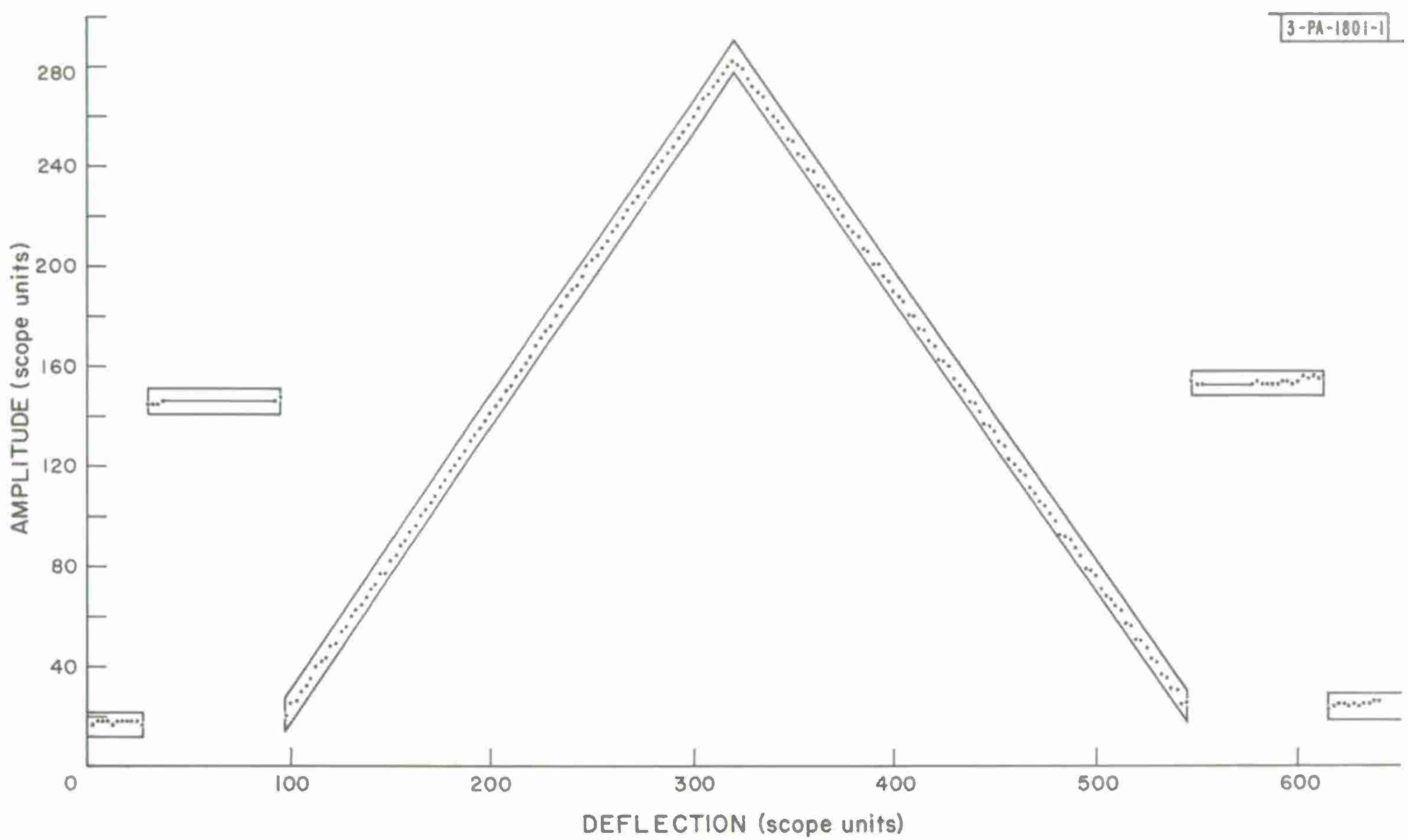


Fig. E-8 Graph of computer output superimposed on projection of film

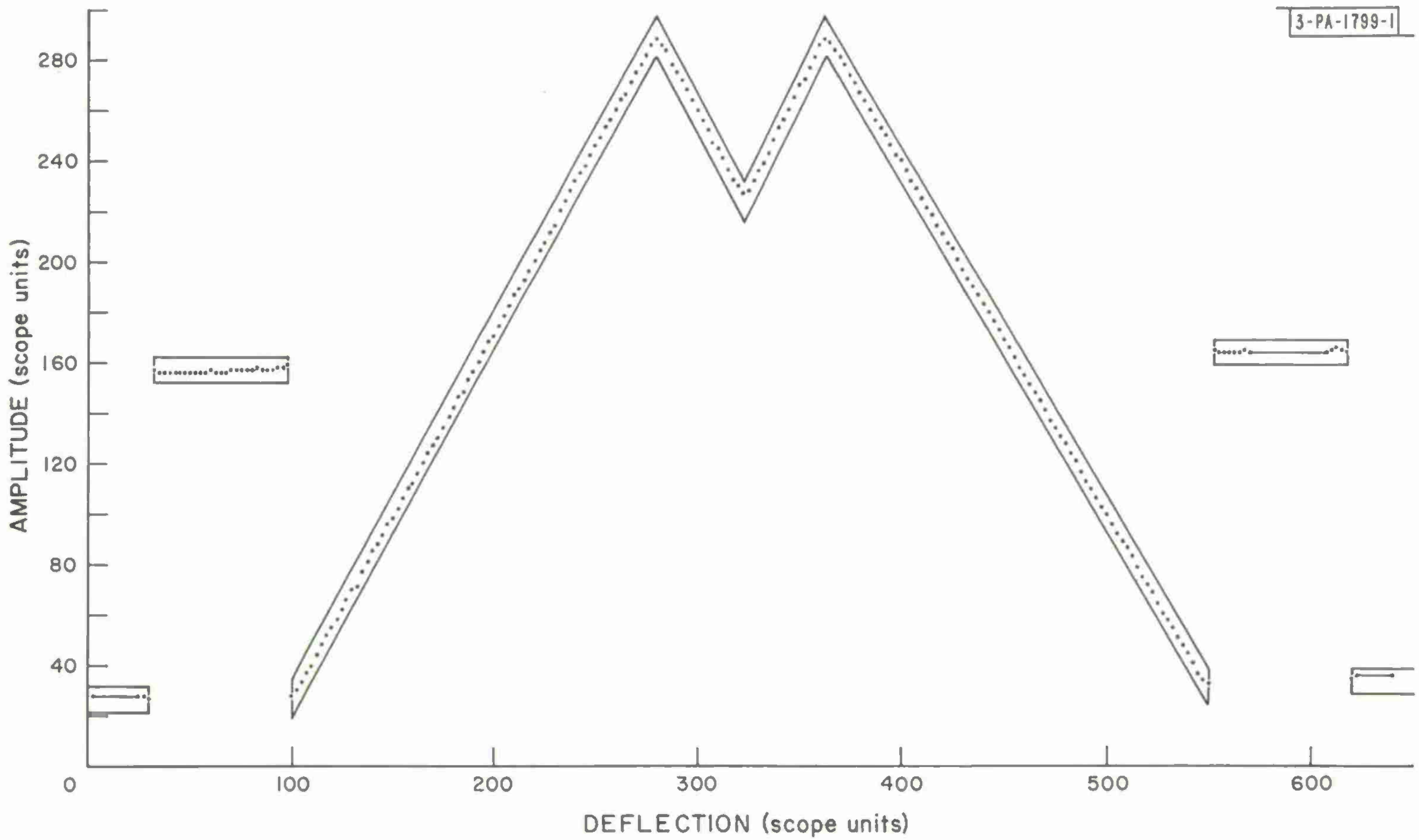


Fig. E-9 Graph of computer output superimposed on projection of film

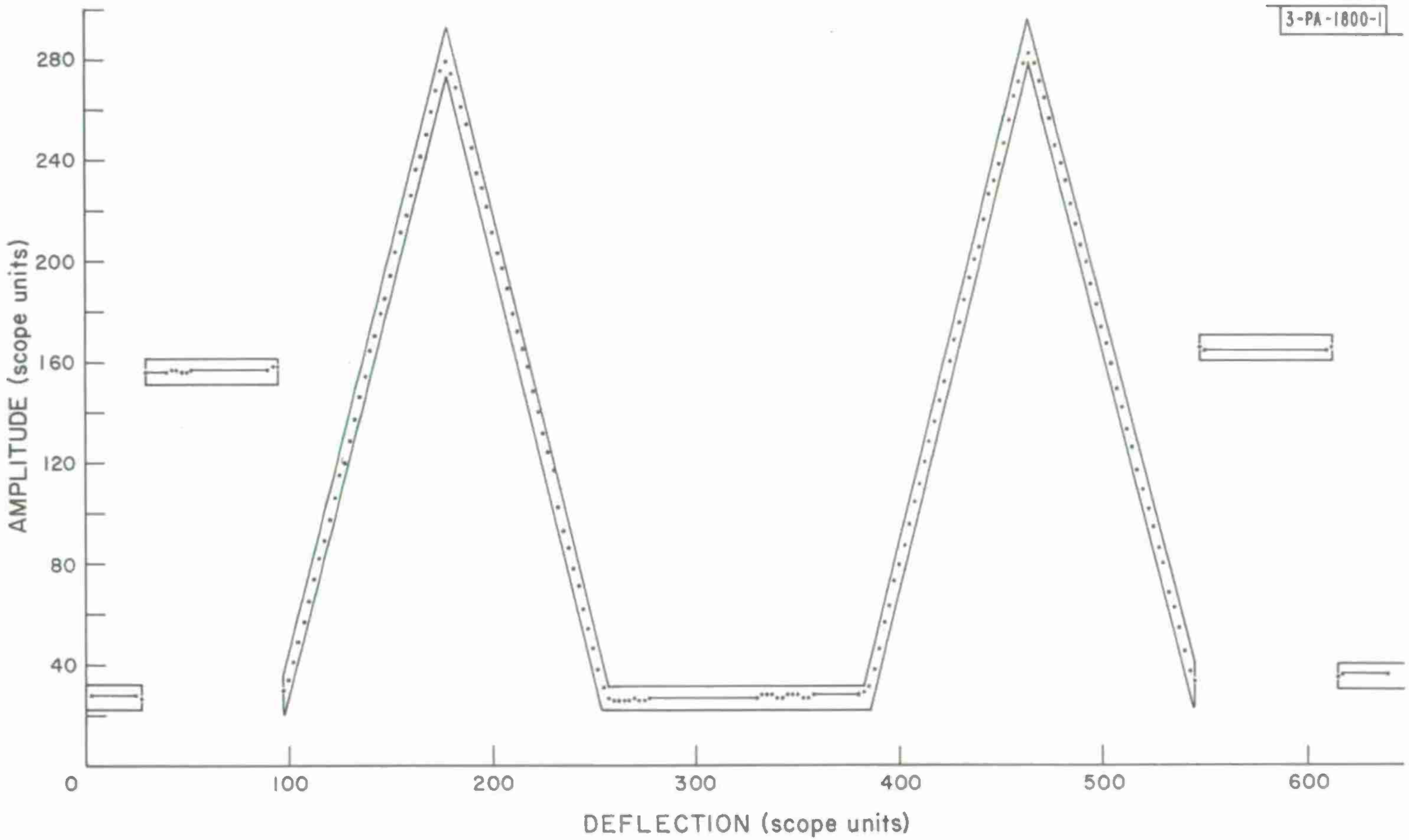


Fig. E-10 Graph of computer output superimposed on projection of film

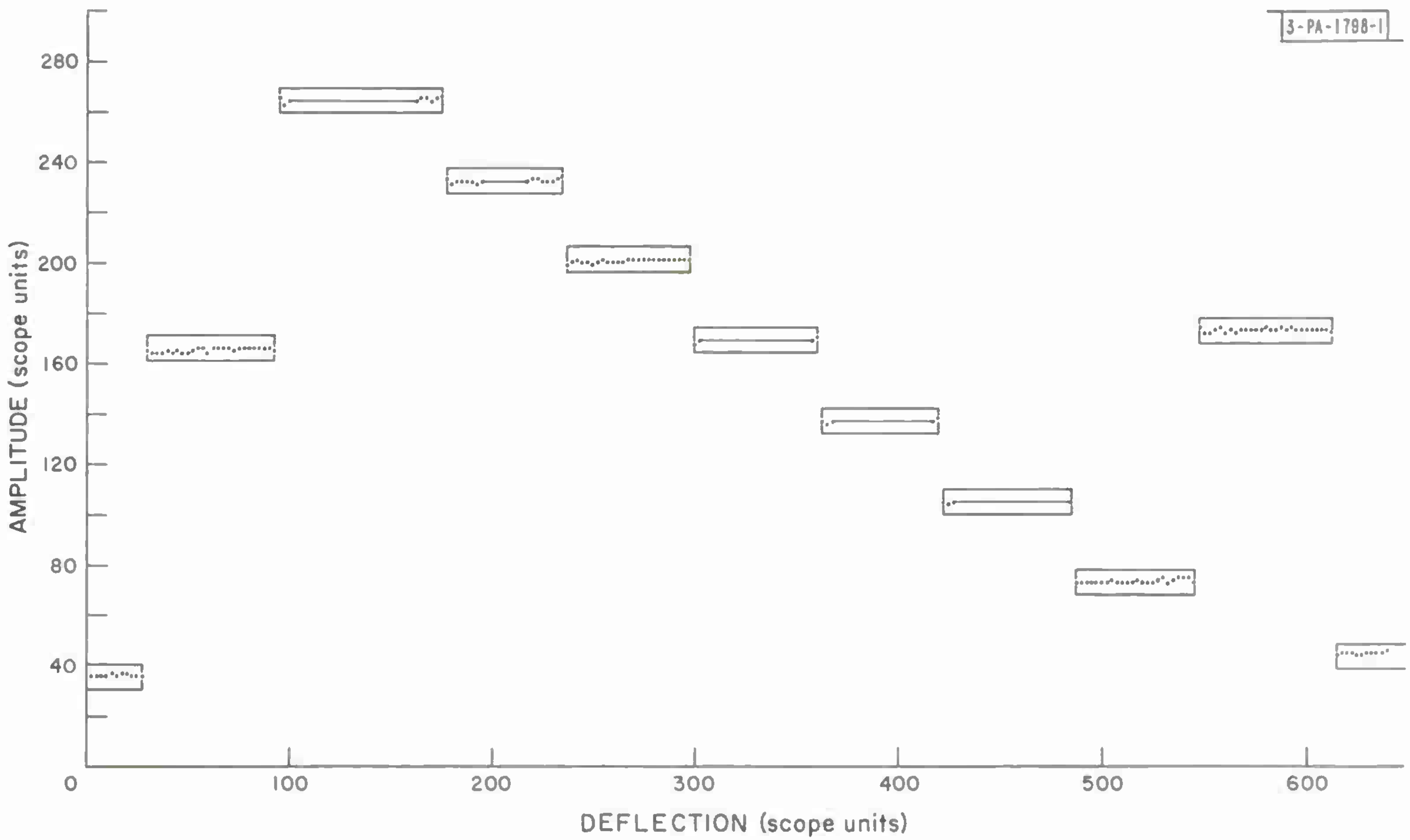


Fig. E-11 Graph of computer output superimposed on projection of film

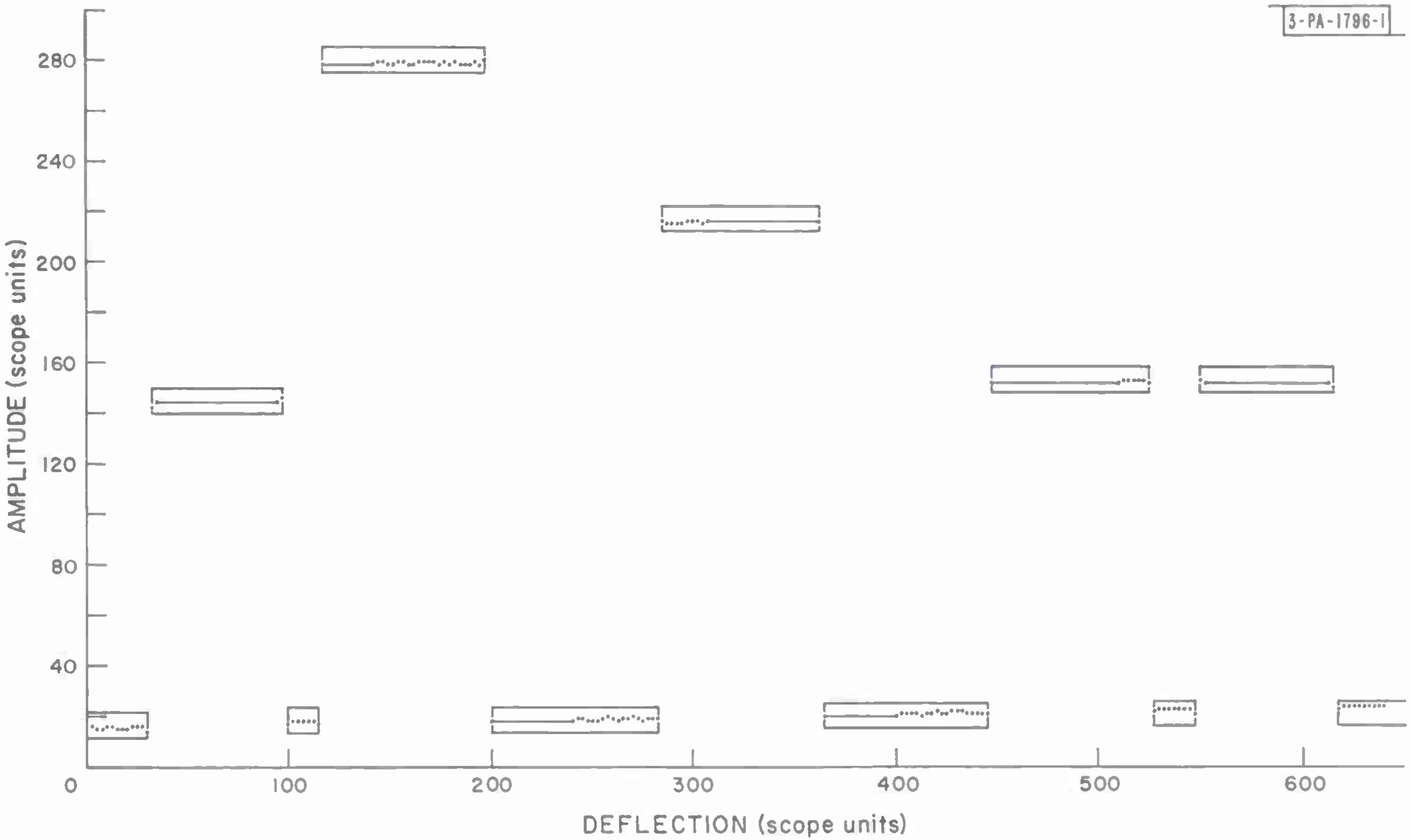


Fig. E-12 Graph of computer output superimposed on projection of film

BIBLIOGRAPHY

1. F-15D PDP-1 Handbook, Digital Equipment Corporation, Maynard, Massachusetts, October 1963.

2. Digital-1-1-S MACRO Assembly Program Manual, Digital Equipment Corporation, Maynard, Massachusetts, 1964.

DOCUMENT CONTROL DATA – R&D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY *(Corporate author)* Lincoln Laboratory, M. I. T.	2a. REPORT SECURITY CLASSIFICATION Unclassified 2b. GROUP

3. REPORT TITLE

Programmable Film Reader

4. DESCRIPTIVE NOTES *(Type of report and inclusive dates)*

Technical Note

5. AUTHOR(S) *(Last name, first name, initial)*

Armenti, A. W., Clapp, D. F., Schulman, A. I., and Wiesen, R. A.

6. REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
10 March 1965	194	2

8a. CONTRACT OR GRANT NO. AF 19(628)-500 b. PROJECT NO. c. d.	9a. ORIGINATOR'S REPORT NUMBER(S) Technical Note 1965-1 9b. OTHER REPORT NO(S) *(Any other numbers that may be assigned this report)* ESD-TDR-65-57

10. AVAILABILITY/LIMITATION NOTICES

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY
	Air Force Systems Command, USAF

13. ABSTRACT

The Programmable Film Reader, consisting of digital computer, magnetic tape units, CRT, and film transport with optical and electronic circuits is a device for reducing radar A-scope film data to digital form. This is done by scanning selected portions of the film with a spot of light under program control. The relative amount of light passing through the film is measured by the device and reported back to the computer for processing.

A set of computer programs, called the Film Reading Programs System, has been written for the PDP-1 computer and Programmable Film Reader. These programs will read films in three formats: A-scope traces, A-scope traces with fiducial marks and Project Radar A-scope traces. The amplitudes of the traces are sampled up to about 500 times and the digitized results written onto magnetic tape in IBM format. This report presents a

(Continued on next page)

14. KEY WORDS

digital computers	film transport	PDP-1	flip-flops
magnetic tape	optical circuits	photomultiplier	filters
cathode ray tubes	electronic circuits	thresholds	

13. ABSTRACT (Continued)

description of the computer programs together with flow charts and listings. The reader is presumed familiar with the PDP-1 computer and the MACRO assembly language.

The latest modification to the system adapts it for use with the MIDAS assembly program and with a new high-speed magnetic tape system on the PDP-1 computer.

Printed by
United States Air Force
L. G. Hanscom Field
Bedford, Massachusetts